

# 10th Innovations in Theoretical Computer Science

ITCS 2019, January 10-12, 2019, San Diego, CA, USA

Edited by  
**Avrim Blum**



*Editor*

Avrim Blum  
Toyota Technological Institute at Chicago (TTIC)  
Chicago, IL, USA  
avrim@ttic.edu

*ACM Classification 2012*

Theory of computation, Mathematics of computing

**ISBN 978-3-95977-095-8**

*Published online and open access by*

Schloss Dagstuhl – Leibniz-Zentrum für Informatik GmbH, Dagstuhl Publishing, Saarbrücken/Wadern, Germany. Online available at <http://www.dagstuhl.de/dagpub/978-3-95977-095-8>.

*Publication date*

January, 2019

*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 <http://dnb.d-nb.de>.

*License*

This work is licensed under a Creative Commons Attribution 3.0 Unported license (CC-BY 3.0): <http://creativecommons.org/licenses/by/3.0/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.2019.0

ISBN 978-3-95977-095-8

ISSN 1868-8969

<http://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)
- Susanne Albers (TU München)
- Christel Baier (TU Dresden)
- Javier Esparza (TU München)
- Michael Mitzenmacher (Harvard University)
- Madhavan Mukund (Chennai Mathematical Institute)
- Anca Muscholl (University Bordeaux)
- Catuscia Palamidessi (INRIA)
- Raimund Seidel (Saarland University and Schloss Dagstuhl – Leibniz-Zentrum für Informatik)
- Thomas Schwentick (TU Dortmund)
- Reinhard Wilhelm (Saarland University)

**ISSN 1868-8969**

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





## ■ Contents

Preface	
<i>Avrim Blum</i> .....	0:ix

### Regular Papers

Submodular Secretary Problem with Shortlists	
<i>Shipra Agrawal, Mohammad Shadravan, and Cliff Stein</i> .....	1:1–1:19
Hamiltonian Sparsification and Gap-Simulation	
<i>Dorit Aharonov and Leo Zhou</i> .....	2:1–2:21
On Solving Linear Systems in Sublinear Time	
<i>Alexandr Andoni, Robert Krauthgamer, and Yosef Poghrow</i> .....	3:1–3:19
Placing Conditional Disclosure of Secrets in the Communication Complexity Universe	
<i>Benny Applebaum and Prashant Nalini Vasudevan</i> .....	4:1–4:14
Bitcoin: A Natural Oligopoly	
<i>Nick Arnosti and S. Matthew Weinberg</i> .....	5:1–5:1
A Simple Sublinear-Time Algorithm for Counting Arbitrary Subgraphs via Edge Sampling	
<i>Sepehr Assadi, Michael Kapralov, and Sanjeev Khanna</i> .....	6:1–6:20
Tensor Network Complexity of Multilinear Maps	
<i>Per Austrin, Petteri Kaski, and Kaie Kubjas</i> .....	7:1–7:21
A #SAT Algorithm for Small Constant-Depth Circuits with PTF Gates	
<i>Swapnam Bajpai, Vaibhav Krishan, Deepanshu Kush, Nutan Limaye, and Srikanth Srinivasan</i> .....	8:1–8:20
Small-Set Expansion in Shortcode Graph and the 2-to-2 Conjecture	
<i>Boaz Barak, Pravesh K. Kothari, and David Steurer</i> .....	9:1–9:12
Algorithms, Bounds, and Strategies for Entangled XOR Games	
<i>Adam Bene Watts, Aram W. Harrow, Gurtej Kanwar, and Anand Natarajan</i> .....	10:1–10:18
Testing Local Properties of Arrays	
<i>Omri Ben-Eliezer</i> .....	11:1–11:20
The Complexity of User Retention	
<i>Eli Ben-Sasson and Eden Saig</i> .....	12:1–12:30
Torus Polynomials: An Algebraic Approach to ACC Lower Bounds	
<i>Abhishek Bhrushundi, Kaave Hosseini, Shachar Lovett, and Sankeerth Rao</i> .....	13:1–13:16
Almost Envy-Free Allocations with Connected Bundles	
<i>Vittorio Bilò, Ioannis Caragiannis, Michele Flammini, Ayumi Igarashi, Gianpiero Monaco, Dominik Peters, Cosimo Vinci, and William S. Zwickler</i> .....	14:1–14:21

10th Innovations in Theoretical Computer Science (ITCS 2019).

Editor: Avrim Blum



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

“Quantum Supremacy” and the Complexity of Random Circuit Sampling <i>Adam Bouland, Bill Fefferman, Chinmay Nirkhe, and Umesh Vazirani</i> .....	15:1–15:2
Adversarially Robust Property-Preserving Hash Functions <i>Elette Boyle, Rio LaVigne, and Vinod Vaikuntanathan</i> .....	16:1–16:20
On Closest Pair in Euclidean Metric: Monochromatic is as Hard as Bichromatic <i>Karthik C. S. and Pasin Manurangsi</i> .....	17:1–17:16
Expander-Based Cryptography Meets Natural Proofs <i>Igor C. Oliveira, Rahul Santhanam, and Roei Tell</i> .....	18:1–18:14
A Note on the Quantum Query Complexity of Permutation Symmetric Functions <i>André Chailloux</i> .....	19:1–19:7
Adaptive Boolean Monotonicity Testing in Total Influence Time <i>Deeparnab Chakrabarty and C. Seshadhri</i> .....	20:1–20:7
On Locality-Sensitive Orderings and Their Applications <i>Timothy M. Chan, Sarel Har-Peled, and Mitchell Jones</i> .....	21:1–21:17
Pseudorandom Generators from the Second Fourier Level and Applications to AC0 with Parity Gates <i>Eshan Chattopadhyay, Pooya Hatami, Shachar Lovett, and Avishay Tal</i> .....	22:1–22:15
Classical Algorithms from Quantum and Arthur-Merlin Communication Protocols <i>Lijie Chen and Ruosong Wang</i> .....	23:1–23:20
Capturing Complementarity in Set Functions by Going Beyond Submodularity/Subadditivity <i>Wei Chen, Shang-Hua Teng, and Hanrui Zhang</i> .....	24:1–24:20
Probabilistic Checking Against Non-Signaling Strategies from Linearity Testing <i>Alessandro Chiesa, Peter Manohar, and Igor Shinkar</i> .....	25:1–25:17
On the Algorithmic Power of Spiking Neural Networks <i>Chi-Ning Chou, Kai-Min Chung, and Chi-Jen Lu</i> .....	26:1–26:20
Last-Iterate Convergence: Zero-Sum Games and Constrained Min-Max Optimization <i>Constantinos Daskalakis and Ioannis Panageas</i> .....	27:1–27:18
Density Estimation for Shift-Invariant Multidimensional Distributions <i>Anindya De, Philip M. Long, and Rocco A. Servedio</i> .....	28:1–28:20
From Local to Robust Testing via Agreement Testing <i>Irit Dinur, Prahladh Harsha, Tali Kaufman, and Noga Ron-Zewi</i> .....	29:1–29:18
Every Set in $\mathcal{P}$ Is Strongly Testable Under a Suitable Encoding <i>Irit Dinur, Oded Goldreich, and Tom Gur</i> .....	30:1–30:17
Alea Iacta Est: Auctions, Persuasion, Interim Rules, and Dice <i>Shaddin Dughmi, David Kempe, and Ruixin Qiang</i> .....	31:1–31:20
Spanoids – An Abstraction of Spanning Structures, and a Barrier for LCCs <i>Zeev Dvir, Sivakanth Gopi, Yuzhou Gu, and Avi Wigderson</i> .....	32:1–32:20

Fairness Under Composition <i>Cynthia Dwork and Christina Ilvento</i> .....	33:1–33:20
A Log-Sobolev Inequality for the Multislice, with Applications <i>Yuval Filmus, Ryan O’Donnell, and Xinyu Wu</i> .....	34:1–34:12
Cubic Formula Size Lower Bounds Based on Compositions with Majority <i>Anna Gál, Avishay Tal, and Adrian Trejo Nuñez</i> .....	35:1–35:13
The Space Complexity of Mirror Games <i>Sumegha Garg and Jon Schneider</i> .....	36:1–36:14
The Subgraph Testing Model <i>Oded Goldreich and Dana Ron</i> .....	37:1–37:19
Adventures in Monotone Complexity and TFNP <i>Mika Göös, Pritish Kamath, Robert Robere, and Dmitry Sokolov</i> .....	38:1–38:19
Algorithmic Polarization for Hidden Markov Models <i>Venkatesan Guruswami, Preetum Nakkiran, and Madhu Sudan</i> .....	39:1–39:19
On the Communication Complexity of Key-Agreement Protocols <i>Iftach Haitner, Noam Mazon, Rotem Oshman, Omer Reingold, and Amir Yehudayoff</i> .....	40:1–40:16
The Paulsen Problem Made Simple <i>Linus Hamilton and Ankur Moitra</i> .....	41:1–41:6
How to Subvert Backdoored Encryption: Security Against Adversaries that Decrypt All Ciphertexts <i>Thibaut Horel, Sunoo Park, Silas Richelson, and Vinod Vaikuntanathan</i> .....	42:1–42:20
On Integer Programming and Convolution <i>Klaus Jansen and Lars Rohwedder</i> .....	43:1–43:17
Empowering the Configuration-IP – New PTAS Results for Scheduling with Setups Times <i>Klaus Jansen, Kim-Manuel Klein, Marten Maack, and Malin Rau</i> .....	44:1–44:19
Being Corrupt Requires Being Clever, But Detecting Corruption Doesn’t <i>Yan Jin, Elchanan Mossel, and Govind Ramnarayan</i> .....	45:1–45:14
Simulating Random Walks on Graphs in the Streaming Model <i>Ce Jin</i> .....	46:1–46:15
On the Complexity of Symmetric Polynomials <i>Markus Bläser and Gorav Jindal</i> .....	47:1–47:14
The Orthogonal Vectors Conjecture for Branching Programs and Formulas <i>Daniel M. Kane and Richard Ryan Williams</i> .....	48:1–48:15
SOS Lower Bounds with Hard Constraints: Think Global, Act Local <i>Pravesh K. Kothari, Ryan O’Donnell, and Tselil Schramm</i> .....	49:1–49:21
Semi-Online Bipartite Matching <i>Ravi Kumar, Manish Purohit, Aaron Schild, Zoya Svitkina, and Erik Vee</i> .....	50:1–50:20

Strategies for Quantum Races <i>Troy Lee, Maharshi Ray, and Miklos Santha</i> .....	51:1–51:21
Lower Bounds for Tolerant Junta and Unateness Testing via Rejection Sampling of Graphs <i>Amit Levi and Erik Waingarten</i> .....	52:1–52:20
Secret Sharing with Binary Shares <i>Fuchun Lin, Mahdi Cheraghchi, Venkatesan Guruswami, Reihaneh Safavi-Naini, and Huaxiong Wang</i> .....	53:1–53:20
On the Communication Complexity of High-Dimensional Permutations <i>Nati Linial, Toniann Pitassi, and Adi Shraibman</i> .....	54:1–54:20
Fisher Zeros and Correlation Decay in the Ising Model <i>Jingcheng Liu, Alistair Sinclair, and Piyush Srivastava</i> .....	55:1–55:8
Quadratic Time-Space Lower Bounds for Computing Natural Functions with a Random Oracle <i>Dylan M. McKay and Richard Ryan Williams</i> .....	56:1–56:20
Random Projection in the Brain and Computation with Assemblies of Neurons <i>Christos H. Papadimitriou and Santosh S. Vempala</i> .....	57:1–57:19
Local Computation Algorithms for Spanners <i>Merav Parter, Ronitt Rubinfeld, Ali Vakilian, and Anak Yodpinyanee</i> .....	58:1–58:21
Proofs of Catalytic Space <i>Krzysztof Pietrzak</i> .....	59:1–59:25
Simple Verifiable Delay Functions <i>Krzysztof Pietrzak</i> .....	60:1–60:15
Sum of Squares Lower Bounds from Symmetry and a Good Story <i>Aaron Potechin</i> .....	61:1–61:20
Learning Time Dependent Choice <i>Zachary Chase and Siddharth Prasad</i> .....	62:1–62:19
Erasures vs. Errors in Local Decoding and Property Testing <i>Sofya Raskhodnikova, Noga Ron-Zewi, and Nithin Varma</i> .....	63:1–63:21
A New Approach to Multi-Party Peer-to-Peer Communication Complexity <i>Adi Rosén and Florent Urrutia</i> .....	64:1–64:19
A Schur Complement Cheeger Inequality <i>Aaron Schild</i> .....	65:1–65:15
Game Efficiency Through Linear Programming Duality <i>Nguyễn Kim Thắng</i> .....	66:1–66:20

## ■ Preface

The papers in this volume were presented at the 10th Innovations in Theoretical Computer Science (ITCS 2019) conference. The conference was held at UC San Diego in San Diego, CA, USA, January 10-12, 2019, and co-located with SODA 2019. 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. The call for papers welcomed all submissions, whether aligned with current theory of computation research directions or deviating from them. A record 202 submissions were received. Of these, the program committee selected 66 papers. I would like to thank the authors of all submissions, whether accepted or not, for their interest in ITCS.

The program committee consisted of 39 members (plus the chair): Scott Aaronson, UT Austin; Eric Blais, Waterloo; Jeremiah Blocki, Purdue; Simina Branzei, Purdue; Bernard Chazelle, Princeton University; Amit Daniely, Hebrew University; Sanjoy Dasgupta, UC San Diego; Zeev Dvir, Princeton University; Uriel Feige, Weizmann; Michal Feldman, Tel-Aviv University; Rong Ge, Duke; Venkatesan Guruswami, CMU; Moritz Hardt, UC Berkeley; Russell Impagliazzo, UC San Diego; Brendan Juba, Washington University St Louis; Varun Kanade, University of Oxford; Eyal Kushilevitz, Technion; Yingyu Liang, University of Wisconsin - Madison; Shachar Lovett, UC San Diego; Sepideh Mahabadi, TTIC; Yishay Mansour, Tel-Aviv University; Rafael Pass, Cornell University; Sofya Raskhodnikova, Boston University; Dana Ron, Tel-Aviv University; Ron Rothblum, Technion; Aviad Rubinfeld, Stanford University; Aaron Sidford, Stanford University; Yaron Singer, Harvard University; Mohit Singh, Georgia Tech; Adam Smith, Boston University; Jacob Steinhardt, UC Berkeley; Madhur Tulsiani, TTIC; Vinod Vaikuntanathan, MIT; Thomas Vidick, Caltech; Matt Weinberg, Princeton University; Ryan Williams, MIT; Mary Wootters, Stanford University; Mihalis Yannakakis, Columbia University; Shengyu Zhang, CUHK and Tencent. I wish to express my heartfelt thanks to them for agreeing to join the committee as well as for investing a great deal of time and effort to evaluate the submissions. I am also grateful to the many subreviewers who assisted with the reviewing process. The local organizer was Shachar Lovett from UC San Diego. I would like to thank him very much for his service. I'm also grateful to Umesh Vazirani, chair of the ITCS Steering Committee, and to Thomas Vidick, who helped with the website among other things. Finally, I would like to thank all the presenters and the audience at ITCS for making ITCS a wonderful experience.

Avrim Blum  
ITCS 2019 Program Chair  
Toyota Technological Institute at Chicago (TTIC)  
Chicago, IL USA



## ITCS 2019 Conference Organization

**Program Chair:** Avrim Blum (TTIC)

**Local Organization:** Shachar Lovett (UC San Diego)

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

**Steering Committee** Sanjeev Arora, Princeton  
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  
Madhu Sudan, Harvard  
Leslie Valiant, Harvard  
Umesh Vazirani, Berkeley  
Thomas Vidick, Caltech  
Avi Wigderson, IAS  
Andy Yao, Tsinghua

**Program Committee:** Scott Aaronson, UT Austin  
Eric Blais, Waterloo  
Jeremiah Blocki, Purdue  
Avrim Blum, TTIC  
Simina Branzei, Purdue  
Bernard Chazelle, Princeton University  
Amit Daniely, Hebrew University  
Sanjoy Dasgupta, UC San Diego  
Zeev Dvir, Princeton University  
Uriel Feige, Weizmann  
Michal Feldman, Tel-Aviv University  
Rong Ge, Duke  
Venkatesan Guruswami, CMU  
Moritz Hardt, UC Berkeley  
Russell Impagliazzo, UC San Diego  
Brendan Juba, Washington University St Louis  
Varun Kanade, University of Oxford  
Eyal Kushilevitz, Technion

**Program Committee** Yingyu Liang, University of Wisconsin - Madison  
**(continued):** Shachar Lovett, UC San Diego  
 Sepideh Mahabadi, TTIC  
 Yishay Mansour, Tel-Aviv University  
 Rafael Pass, Cornell University  
 Sofya Raskhodnikova, Boston University  
 Dana Ron, Tel-Aviv University  
 Ron Rothblum, Technion  
 Aviad Rubinfeld, Stanford University  
 Aaron Sidford, Stanford University  
 Yaron Singer, Harvard University  
 Mohit Singh, Georgia Tech  
 Adam Smith, Boston University  
 Jacob Steinhardt, UC Berkeley  
 Madhur Tulsiani, TTIC  
 Vinod Vaikuntanathan, MIT  
 Thomas Vidick, Caltech  
 Matt Weinberg, Princeton University  
 Ryan Williams, MIT  
 Mary Wootters, Stanford University  
 Mihalis Yannakakis, Columbia University  
 Shengyu Zhang, CUHK and Tencent

<b>Additional</b>	Maryam Aliakbarpour	Jonathan Allcock	Josh Alman
<b>Reviewers:</b>	Sepehr Assadi	Miriam Backens	Arturs Backurs
	Eric Balkanski	Marshall Ball	Paul Beame
	Xiaohui Bei	Alexander Belov	Hedyeh Beyhaghi
	Arnab Bhattacharyya	Alexander Block	Andrej Bogdanov
	Joshua Brody	Niv Buchbinder	Yang Cai
	Clement Canonne	T-H. Hubert Chan	Eshan Chattopadhyay
	Lijie Chen	Yu Cheng	Bram Cohen
	Ran Cohen	Vincent Cohen-Addad	Yuval Dagan
	Sarah Dean	Holger Dell	Travis Dick
	Dean Doron	Andy Drucker	Talya Eden
	Faith Ellen	Meryem Essaidi	Tomer Ezra
	Arman Fazeli	Bill Fefferman	Zhe Feng
	Yuval Filmus	Aris Filos-Ratsikas	Ophir Friedler
	Hu Fu	Jugal Garg	Michal Garlik
	Kira Goldner	Mika Goos	Fernando Jeronimo
	Fred Green	Elena Grigorescu	Joshua Grochow
	Siyao Guo	Avinatan Hassidim	Matt Hastings
	Pooya Hatami	Niao He	John Hitchcock
	Jan Hladky	Samuel Hopkins	Kaave Hosseini

**Additional Reviewers (continued):**

Justin Hsu	Rahul Jain	Dimitris Kalimeris
Daniel Kane	Thomas Kesselheim	Hartmut Klauck
Shimon Kogan	Ilan Komargodski	Tomer Koren
Elias Koutsoupias	Robert Krauthgamer	Janardhan Kulkarni
Akash Kumar	Matt Kusner	Lap Chi Lau
Vedat Levi Alev	Jian Li	Ben Liao
Lydia Liu	Zhenming Liu	Pinyan Lu
Mohammad Mahmoody	Hemanta Maji	Frederik Mallmann-Trenn
Jieming Mao	Andrew McGregor	John Miller
Dor Minzer	Slobodan Mitrovic	Divyarthi Mohan
Ryuhei Mori	Cameron Musco	Hoi Nguyen
Aleksandar Nikolov	Katarzyna Paluch	Denis Pankratov
Eric Price	Christos-Alexandros Psomas	Sharon Qian
Miklos Z. Racz	Manish Raghavan	Govind Ramnarayan
Dror Rawitz	Alireza Rezaei	Andrea Rocchetto
Liam Roditty	Will Rosenbaum	Nir Rosenfeld
Adi Rosen	Guy Rothblum	Tim Roughgarden
Shubhangi Saraf	Tselil Schramm	Ariel Schwartzman
Kineret Segal	Siddhartha Sen	C. Seshadhri
Amirbehshad Shahrashbi	Asaf Shapira	Max Simchowitz
Sahil Singla	Christian Sohler	David Soloveichik
Yonatan Sompolinsky	Vasilis Syrgkanis	Avishay Tal
Inbal Talgam-Cohen	Nirvan Tyagi	Falk Unger
Ali Vakilian	Shai Vardi	Nithin Varma
Virginia Vassilevska Williams	Suresh Venkatasubramanian	Matheus Venturynne
Erik Waingarten	Omri Weinstein	John Wright
Ning Xie	Lin Yang	Junjie Ye
Amir Yehudayoff	Anak Yodpinyanee	Eylon Yogev
Henry Yuen	Chihao Zhang	Hongyang Zhang
Jiapeng Zhang	Samson Zhou	Vassilis Zikas
Tijana Zrnic		



# Submodular Secretary Problem with Shortlists

Shipra Agrawal<sup>1</sup>

Columbia University, New York, NY, USA, 10027  
sa3305@columbia.edu

Mohammad Shadravan

Columbia University, New York, NY, USA, 10027  
ms4961@columbia.edu

Cliff Stein<sup>2</sup>

Columbia University, New York, NY, USA, 10027  
cliff@ieor.columbia.edu

---

## Abstract

---

In submodular  $k$ -secretary problem, the goal is to select  $k$  items in a randomly ordered input so as to maximize the expected value of a given monotone submodular function on the set of selected items. In this paper, we introduce a relaxation of this problem, which we refer to as submodular  $k$ -secretary problem with shortlists. In the proposed problem setting, the algorithm is allowed to choose more than  $k$  items as part of a shortlist. Then, after seeing the entire input, the algorithm can choose a subset of size  $k$  from the bigger set of items in the shortlist. We are interested in understanding to what extent this relaxation can improve the achievable competitive ratio for the submodular  $k$ -secretary problem. In particular, using an  $O(k)$  sized shortlist, can an online algorithm achieve a competitive ratio close to the best achievable offline approximation factor for this problem? We answer this question affirmatively by giving a polynomial time algorithm that achieves a  $1 - 1/e - \epsilon - O(k^{-1})$  competitive ratio for any constant  $\epsilon > 0$ , using a shortlist of size  $\eta_\epsilon(k) = O(k)$ . This is especially surprising considering that the best known competitive ratio (in polynomial time) for the submodular  $k$ -secretary problem is  $(1/e - O(k^{-1/2}))(1 - 1/e)$  [20].

The proposed algorithm also has significant implications for another important problem of submodular function maximization under random order streaming model and  $k$ -cardinality constraint. We show that our algorithm can be implemented in the streaming setting using a memory buffer of size  $\eta_\epsilon(k) = O(k)$  to achieve a  $1 - 1/e - \epsilon - O(k^{-1})$  approximation. This result substantially improves upon [28], which achieved the previously best known approximation factor of  $1/2 + 8 \times 10^{-14}$  using  $O(k \log k)$  memory; and closely matches the known upper bound for this problem [24].

**2012 ACM Subject Classification** Mathematics of computing → Submodular optimization and polymatroids, Theory of computation → Online algorithms, Theory of computation → Streaming, sublinear and near linear time algorithms

**Keywords and phrases** Submodular Optimization, Secretary Problem, Streaming Algorithms

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2019.1

**Related Version** A full version of the paper is available as [1], <https://arxiv.org/abs/1809.05082>.

---

<sup>1</sup> Research supported in part by Google Faculty Research Awards 2017 and Amazon Research Awards 2017.

<sup>2</sup> Research supported in part by NSF grants CCF-1421161 and CCF-1714818.



## 1 Introduction

In the classic *secretary problem*,  $n$  items appear in random order. We know  $n$ , but don't know the value of an item until it appears. Once an item arrives, we have to irrevocably and immediately decide whether or not to select it. Only one item is allowed to be selected, and the objective is to select the most valuable item, or perhaps to maximize the expected value of the selected item [11, 15, 23]. It is well known that the optimal policy is to observe the first  $n/e$  items without making any selection and then select the first item whose value is larger than the value of the best item in the first  $n/e$  items [11]. This algorithm, given by [11], is asymptotically optimal, and hires the best secretary with probability at least  $1/e$ . Hence it is also  $1/e$ -competitive for the expected value of the chosen item, and it can be shown that no algorithm can beat  $1/e$ -competitive ratio in expectation.

Many variants and generalizations of the secretary problem have been studied in the literature, see e.g., [3, 32, 30, 33, 21, 4]. [21, 4] introduced a multiple choice secretary problem, where the goal is to select  $k$  items in a randomly ordered input so as to maximize the *sum* of their values; and [21] gave an algorithm with an asymptotic competitive ratio of  $1 - O(1/\sqrt{k})$ . Thus as  $k \rightarrow \infty$ , the competitive ratio approaches 1. Recent literature studied several generalizations of this setting to multidimensional knapsacks [26], and proposed algorithms for which the expected online solution approaches the best offline solution as the knapsack sizes become large (e.g., [13, 10, 2]).

In another variant of multiple-choice secretary problem, [6] and [16] introduce the *submodular  $k$ -secretary problem*. In this secretary problem, the algorithm again selects  $k$  items, but the value of the selected items is given by a monotone submodular function  $f$ . The algorithm has value oracle access to the function, i.e., for any given set  $T$ , an algorithm can query an oracle to find its value  $f(T)$  [31]. The algorithm can select at most  $k$  items,  $a_1 \cdots, a_k$ , from a randomly ordered sequence of  $n$  items. The goal is to maximize  $f(\{a_1, \dots, a_k\})$ . Currently, the best result for this setting is due to [20], who achieve a  $1/e$ -competitive ratio in exponential time, or  $\frac{1}{e}(1 - \frac{1}{e})$  in polynomial time. In this case, the offline problem is NP-hard and hard-to approximate beyond the factor of  $1 - 1/e$  achieved by the greedy algorithm [27]. However, it is unclear if a competitive ratio of  $1 - 1/e$  can be achieved by an online algorithm for the submodular  $k$ -secretary problem even when  $k$  is large.

### Our model: secretary problem with shortlists

In this paper, we consider a relaxation of the secretary problem where the algorithm is allowed to select a *shortlist* of items that is larger than the number of items that ultimately need to be selected. That is, in a multiple-choice secretary problem with cardinality constraint  $k$ , the algorithm is allowed to choose more than  $k$  items as part of a shortlist. Then, after seeing the entire input, the algorithm can choose a subset of size  $k$  from the bigger set of items in the shortlist.

This new model is motivated by some practical applications of secretary problems, such as hiring (or assignment problems), where in some cases it may be possible to tentatively accept a larger number of candidates (or requests), while deferring the choice of the final  $k$ -selections to after all the candidates have been seen. Since there may be a penalty for declining candidates who were part of the shortlist, one would prefer that the shortlist is not much larger than  $k$ .

Another important motivation is theoretical: we wish to understand to what extent this relaxation of the secretary problem can improve the achievable competitive ratio. This question is in the spirit of several other methods of analysis that allow an online algorithm to have additional power, such as *resource augmentation* [18, 29].

The potential of this relaxation is illustrated by the basic secretary problem, where the aim is to select the item of maximum value among randomly ordered inputs. There, it is not difficult to show that if an algorithm picks every item that is better than the items seen so far, the true maximum will be found, while the expected number of items picked under randomly ordered inputs will be  $O(\log n)$ . Further, we show that this approach can be easily modified to get the maximum with  $1 - \epsilon$  probability while picking at most  $O(\ln(1/\epsilon))$  items for any constant  $\epsilon > 0$ . Thus, with just a constant sized shortlist, we can break the  $1/e$  barrier for the secretary problem and achieve a competitive ratio that is arbitrarily close to 1.

Motivated by this observation, we ask if a similar improvement can be achieved by relaxing the submodular  $k$ -secretary problem to allow a shortlist. That is, instead of choosing  $k$  items, the algorithm is allowed to choose  $\eta(k)$  items as part of a shortlist, for some function  $\eta$ ; and at the end of all inputs, the algorithm chooses  $k$  items from the  $\eta(k)$  selected items. Then, what is the relationship between  $\eta(\cdot)$  and the competitive ratio for this problem? Can we achieve a solution close to the best offline solution when  $\eta(k)$  is not much bigger than  $k$ , for example when  $\eta(k) = \theta(k)$ ?

In this paper, we answer this question affirmatively by giving a polynomial time algorithm that achieves  $1 - 1/e - \epsilon - O(k^{-1})$  competitive ratio for the submodular  $k$ -secretary problem using a shortlist of size  $\eta(k) = O(k)$ . This is surprising since  $1 - 1/e$  is the best achievable approximation (in polynomial time) for the offline problem. Further, for some special cases of submodular functions, we demonstrate that an  $O(1)$  shortlist allows us to achieve a  $1 - \epsilon$  competitive ratio. These results demonstrate the power of (small) shortlists for closing the gap between online and offline (polynomial time) algorithms.

We also discuss connections of secretary problem with shortlists to the related streaming settings. While a streaming algorithm does not qualify as an online algorithm (even when a shortlist is allowed), we show that our algorithm can in fact be implemented in a streaming setting to use  $\eta(k) = O(k)$  memory buffer; and our results significantly improve the available results for the submodular random order streaming problem.

## 1.1 Problem Definition

We now give a more formal definition. Items from a set  $\mathcal{U} = \{a_1, a_2, \dots, a_n\}$  (pool of items) arrive in a uniformly random order over  $n$  sequential rounds. The set  $\mathcal{U}$  is a priori fixed but unknown to the algorithm, and the total number of items  $n$  is known to the algorithm. In each round, the algorithm irrevocably decides whether to add the arriving item to a *shortlist*  $A$  or not. The algorithm's value at the end of  $n$  rounds is given by

$$\text{ALG} = \mathbb{E}[\max_{S \subseteq A, |S| \leq k} f(S)]$$

where  $f(\cdot)$  is a monotone submodular function. The algorithm has value oracle access to this function. The optimal offline utility is given by

$$\text{OPT} := f(S^*), \text{ where } S^* = \arg \max_{S \subseteq [n], |S| \leq k} f(S).$$

We say that an algorithm for this problem achieves a competitive ratio  $c$  using shortlist of size  $\eta(k)$ , if at the end of  $n$  rounds,  $|A| \leq \eta(k)$  and  $\frac{\text{ALG}}{\text{OPT}} \geq c$ .

Given the shortlist  $A$ , since the problem of computing the solution  $\arg \max_{S \subseteq A, |S| \leq k} f(S)$  can itself be computationally intensive, our algorithm will also track and output a subset  $A^* \subseteq A, |A^*| \leq k$ . We will lower bound the competitive ratio by bounding  $\frac{f(A^*)}{f(S^*)}$ .

The above problem definition has connections to some existing problems studied in the literature. The well-studied online submodular  $k$ -secretary problem described earlier is obtained from the above definition by setting  $\eta(k) = k$ , i.e., it is same as the case when no extra items can be selected as part of a shortlist. Another related problem is *submodular random order streaming problem* studied in [28]. In that problem, items from a set  $\mathcal{U}$  arrive online in random order and the algorithm aims to select a subset  $S \subseteq \mathcal{U}$ ,  $|S| \leq k$  in order to maximize  $f(S)$ . The streaming algorithm is allowed to maintain a *buffer* of size  $\eta(k) \geq k$ . However, the streaming problem is distinct from the submodular  $k$ -secretary problem with shortlists in several important ways. On one hand, since an item previously selected in the memory buffer can be discarded and replaced by a new items, a memory buffer of size  $\eta(k)$  does not imply a shortlist of size at most  $\eta(k)$ . On the other hand, in the secretary setting, we are allowed to memorize/store more than  $\eta(k)$  items without adding them to the shortlist. Thus an algorithm for submodular  $k$ -secretary problem with shortlist of size  $\eta(k)$  may potentially use a buffer of size larger than  $\eta(k)$ . Our algorithms, as described in the paper, do use a large buffer. But we will show those algorithms can in fact be implemented to use only  $\eta(k) = O(k)$  buffer, thus obtaining matching results for the streaming problem.

## 1.2 Our Results

Our main contribution is an online algorithm for the submodular  $k$ -secretary problem with shortlists that, for any constant  $\epsilon > 0$ , achieves a competitive ratio of  $1 - \frac{1}{e} - \epsilon - O(\frac{1}{k})$  with  $\eta(k) = O(k)$ . Note that for submodular  $k$ -secretary problem there is an upper bound of  $1 - 1/e$  on the achievable approximation factor, even in the offline setting, and this upper bound applies to our problem for arbitrary size  $\eta(\cdot)$  of shortlists. On the other hand for online monotone submodular  $k$ -secretary problem, i.e., when  $\eta(k) = k$ , the best competitive ratio achieved in the literature is  $1/e - O(k^{-1/2})$  [20]. Remarkably, with only an  $O(k)$  size shortlist, our online algorithm is able to achieve a competitive ratio that is arbitrarily close to the offline upper bound of  $1 - 1/e$ .

In the theorem statements below, big-Oh notation  $O(\cdot)$  is used to represent asymptotic behavior with respect to  $k$  and  $n$ . We assume the standard value oracle model: the only access to the submodular function is through a black box returning  $f(S)$  for a given set  $S$ , and each such query can be done in  $O(1)$  time.

► **Theorem 1.** *For any constant  $\epsilon > 0$ , there exists an online algorithm (Algorithm 2) for the submodular  $k$ -secretary problem with shortlists that achieves a competitive ratio of  $1 - \frac{1}{e} - \epsilon - O(\frac{1}{k})$ , with shortlist of size  $\eta_\epsilon(k) = O(k)$ . Here,  $\eta_\epsilon(k) = O(2^{\text{poly}(1/\epsilon)} k)$ .*

Specifically, we have  $\eta_\epsilon(k) = c \frac{\log(1/\epsilon)}{\epsilon^2} \left( \frac{1}{\epsilon^6} \frac{\log(1/\epsilon)}{\epsilon^4 \log(1/\epsilon)} \right) k$  for some constant  $c$ .

Further, we give an efficient implementation of Algorithm 2 that uses a memory buffer of size at most  $\eta_\epsilon(k)$  to get the following result for the problem of *submodular random order streaming problem* described in the previous section.

► **Theorem 2.** *For any constant  $\epsilon \in (0, 1)$ , there exists an algorithm for the submodular random order streaming problem that achieves  $1 - \frac{1}{e} - \epsilon - O(\frac{1}{k})$  approximation to  $OPT$  while using a memory buffer of size at most  $\eta_\epsilon(k) = O(k)$ . Also, the number of objective function evaluations for each item, amortized over  $n$  items, is  $O(1 + \frac{k^2}{n})$ .*

The above result significantly improves over the state-of-the-art results in random order streaming model [28], which are an approximation ratio of  $\frac{1}{2} + 8 \times 10^{-14}$  using a memory of size  $O(k \log k)$ . In addition it closely matches the known upper bound for this problem [24].

In [24], the authors demonstrate the existence of a monotone submodular function  $f$  such that any constant-pass algorithm that finds a  $(1+\epsilon)(1-1/k)^k$  approximation with probability at least 0.99 requires  $\Omega(n/k^2)$  space in random order streaming model.

Also note from Theorem 2 that our algorithm can be implemented with running time linear in  $n$ , the size of the input ( $O(n+k^2)$  time to be precise). This is significant as, until recently, it was not known if there exists a linear time algorithm achieving a  $1-1/e-\epsilon$  approximation even for the offline monotone submodular maximization problem under cardinality constraint [25]. Another interesting aspect of our algorithm is that it is highly parallel. Even though the decision for each arriving item may take time that is exponential in  $1/\epsilon$  (roughly  $\eta_\epsilon(k)/k$ ), it can be readily parallelized among multiple (as many as  $\eta_\epsilon(k)/k$ ) processors.

It is natural to ask whether these shortlists are, in fact, too powerful. Maybe they could actually allow us to always match the best offline algorithm. We give a negative result in this direction and show that even if we have unlimited computation power, for any function  $\eta(k) = o(n)$ , we can get no better than  $7/8$ -competitive algorithm using a shortlist of size  $\eta(k)$ . Note that with unlimited computational power, the offline problem can be solved exactly. This result demonstrates that having a shortlist does not make the online problem too easy - even with a shortlist (of size  $o(n)$ ) there is an information theoretic gap between the online and offline problem.

► **Theorem 3.** *No online algorithm (even with unlimited computational power) can achieve a competitive ratio better than  $7/8 + o(1)$  for the submodular  $k$ -secretary problem with shortlists, while using a shortlist of size  $\eta(k) = o(n)$ .*

Finally, for some special cases of monotone submodular functions, we can asymptotically approach the optimal solution. The first one is the family of functions we call  $m$ -submodular. A function  $f$  is  $m$ -submodular if it is submodular and there exists a submodular function  $F$  such that for all  $S$ :

$$f(S) = \max_{T \subseteq S, |T| \leq m} F(T).$$

► **Theorem 4.** *If  $f$  is an  $m$ -submodular function, there exists an online algorithm for the submodular  $k$ -secretary problem with shortlists that achieves a competitive ratio of  $1-\epsilon$  with shortlist of size  $\eta_{\epsilon,m}(k) = O(1)$ . Here,  $\eta_{\epsilon,m}(k) = (2m+3) \ln(2/\epsilon)$ .*

A proof of Theorem 4 along with the relevant algorithm appear in the full version [1].

Another special case is monotone submodular functions  $f$  satisfying the following property:  $f(\{a_1, \dots, a_i + \alpha, \dots, a_k\}) \geq f(\{a_1, \dots, a_i, \dots, a_k\})$ , for any  $\alpha > 0$  and  $1 \leq i \leq k$ . We can show that the algorithm by [21] asymptotically approaches optimal solution for such functions, but we omit the details.

### 1.3 Comparison to related work

We compare our results (Theorem 1 and Theorem 2) to the best known results for *submodular  $k$ -secretary problem* and *submodular random order streaming problem*, respectively.

The best known algorithm so far for submodular  $k$ -secretary problem is by [20], with asymptotic competitive ratio of  $1/e - O(k^{-1/2})$ . In their algorithm, after observing each element, they use an oracle to compute optimal offline solution on the elements seen so far. Therefore it requires exponential time in  $n$ . The best competitive ratio that they can get in polynomial time is  $\frac{1}{e}(1 - \frac{1}{e}) - O(k^{-1/2})$ . In comparison, by using a shortlist of size  $O(k)$  our

■ **Table 1** submodular  $k$ -secretary problem settings.

	#selections	Comp ratio	Running time	Comp ratio in poly( $n$ )
[20]	$k$	$1/e - O(k^{-1/2})$	$exp(n)$	$\frac{1}{e}(1 - 1/e)$
this	$O_\epsilon(k)$	$1 - 1/e - \epsilon - O(1/k)$	$O_\epsilon(n)$	$1 - 1/e - \epsilon - O(1/k)$

■ **Table 2** submodular random order streaming problem.

	Memory size	Approximation ratio	Running time	update time
[17]	$O(k)$	0.19	$O(n)$	$O(1)$
[28]	$O(k \log k)$	$1/2 + 8 \times 10^{-14}$	$O(n \log k)$	$O(\log k)$
[5]	$O(\frac{1}{\epsilon} k \log k)$	$1/2 - \epsilon$	$poly(n, k, 1/\epsilon)$	$O(\frac{1}{\epsilon} \log k)$
this	$O_\epsilon(k)$	$1 - 1/e - \epsilon - O(1/k)$	$O_\epsilon(n + k^2)$	amortized $O_\epsilon(1 + \frac{k^2}{n})$

(polynomial time) algorithm achieves a competitive ratio of  $1 - \frac{1}{e} - \epsilon - O(k^{-1})$ . In addition to substantially improving the above-mentioned result for submodular  $k$ -secretary problem, this closely matches the best possible offline approximation ratio of  $1 - 1/e$  in polynomial time. Further, our algorithm is linear time. Table 1 summarizes this comparison. Here,  $O_\epsilon(\cdot)$  hides the dependence on the constant  $\epsilon$ . The hidden constant in  $O_\epsilon(\cdot)$  is  $c \frac{\log(1/\epsilon)}{\epsilon^2} \left( \frac{1}{\epsilon^6} \log(1/\epsilon) \right)^{\frac{1}{\epsilon^4} \log(1/\epsilon)}$  for some absolute constant  $c$ .

In the streaming setting, [9] provided a single pass streaming algorithm for monotone submodular function maximization under  $k$ -cardinality constraint, that achieves a 0.25 approximation under adversarial ordering of input. Their algorithm requires  $O(1)$  function evaluations per arriving item and  $O(k)$  memory. The currently best known approximation under adversarial order streaming model is by [5], who achieve a  $1/2 - \epsilon$  approximation with a memory of size  $O(\frac{1}{\epsilon} k \log k)$ . There is an  $1/2 + o(1)$  upper bound on the competitive ratio achievable by any streaming algorithm for submodular maximization that only queries the value of the submodular function on feasible sets (i.e., sets of cardinality at most  $k$ ) while using  $o(n)$  memory [28].

[17] initiated the study of submodular random order streaming problem. Their algorithm uses  $O(k)$  memory and a total of  $n$  function evaluations to achieve 0.19 approximation. The state of the art result in the random order input model is due to [28] who achieve a  $1/2 + 8 \times 10^{-14}$  approximation, while using a memory buffer of size  $O(k \log k)$ .

Table 2 provides a detailed comparison of our result in Theorem 2 to the above-mentioned results for submodular random order streaming problem, showing that our algorithm substantially improves the existing results for most aspects of the problem.

There is also a line of work studying the online variant of the submodular welfare maximization problem (e.g., [22, 7, 19]). In this problem, the items arrive online, and each arriving item should be allocated to one of  $m$  agents with a submodular valuation functions  $w_i(S_i)$  where  $S_i$  is the subset of items allocated to  $i$ -th agent). The goal is to partition the arriving items into  $m$  sets to be allocated to  $m$  agents, so that the sum of valuations over all agents is maximized. This setting is incomparable with the submodular  $k$ -secretary problem setting considered here.

## 1.4 Organization

The rest of the paper is organized as follows. Section 2 describes our main algorithm (Algorithm 2) for the submodular  $k$ -secretary problem with shortlists, and demonstrates that its shortlist size is bounded by  $\eta_\epsilon(k) = O(k)$ . In Section 3, we analyze the competitive ratio

---

**Algorithm 1** Algorithm for secretary with shortlist. (finding max online)
 

---

```

1: Inputs: number of items  $N$ , items in  $I = \{a_1, \dots, a_N\}$  arriving sequentially,  $\delta \in (0, 1]$ .
2: Initialize:  $A \leftarrow \emptyset$ ,  $u = n\delta/2$ ,  $M = -\infty$ 
3:  $L \leftarrow 4 \ln(2/\delta)$ 
4: for  $i = 1$  to  $N$  do
5:   if  $a_i > M$  then
6:      $M \leftarrow a_i$ 
7:     if  $i \geq u$  and  $|A| < L$  then
8:        $A \leftarrow A \cup \{a_i\}$ 
9:     end if
10:  end if
11: end for
12: return  $A$ , and  $A^* := \max_{i \in A} a_i$ 

```

---

of this algorithm to prove Theorem 1. In Section 4, we provide an alternate implementation of Algorithm 2 that uses a memory buffer of size at most  $\eta_\epsilon(k)$ , in order to prove Theorem 2. Finally, in Section 5, we provide a proof of our impossibility result stated in Theorem 3. The proof of Theorem 4 along with the relevant algorithm can be found in the full version [1].

## 2 Algorithm description

Before giving our algorithm for submodular  $k$ -secretary problem with shortlists, we describe a simple technique for (classic) secretary problem with shortlists that achieves a  $1 - \delta$  competitive ratio using shortlists of size logarithmic in  $1/\delta$ . Recall that in the secretary problem, the aim is to select an item with expected value close to the maximum among a pool of items  $I = (a_1, \dots, a_N)$  arriving sequentially in a uniformly random order. We will consider the variant with shortlists, where we now want to pick a shortlist which contains an item with expected value close to the maximum. We propose the following simple algorithm. For the first  $n\delta/2$  rounds, don't add any items to the shortlist, but just keep track of the maximum value seen so far. For all subsequent rounds, for any arriving item  $i$  that has a value  $a_i$  greater than or equal to the maximum value seen so far, add it to the shortlist if number of items added so far is less than or equal to  $L = 4 \ln(2/\delta)$ . This algorithm is summarized as Algorithm 1. Clearly, for constant  $\delta$ , this algorithm uses a shortlist of size  $L = O(1)$ . Further, under a uniform random ordering of input, we can show that the maximum value item will be part of the shortlist with probability  $1 - \delta$ . (See Proposition 25 in Section 3.)

There are two main difficulties in extending this idea to the submodular  $k$ -secretary problem with shortlists. First, instead of one item, here we aim to select a set  $S$  of  $k$  items using an  $O(k)$  length shortlist. Second, the contribution of each new item  $i$  to the objective value, as given by the submodular function  $f$ , depends on the set of items selected so far.

The first main concept we introduce to handle these difficulties is that of dividing the input into sequential blocks that we refer to as  $(\alpha, \beta)$  windows. Below is the precise construction of  $(\alpha, \beta)$  windows, for any positive integers  $\alpha$  and  $\beta$ , such that  $k/\alpha$  is an integer.

We use a set of random variables  $X_1, \dots, X_m$  defined in the following way. Throw  $n$  balls into  $m$  bins uniformly at random. Then set  $X_j$  to be the number of balls in the  $j$ th bin. We call the resulting  $X_j$ 's a  $(n, m)$ -ball-bin random set.



---

**Algorithm 2** Algorithm for submodular  $k$ -secretary with shortlist.
 

---

- 1: Inputs: set  $\bar{I} = \{\bar{a}_1, \dots, \bar{a}_n\}$  of  $n$  items arriving sequentially, submodular function  $f$ , parameter  $\epsilon \in (0, 1]$ .
  - 2: Initialize:  $S_0 \leftarrow \emptyset, R_0 \leftarrow \emptyset, A \leftarrow \emptyset, A^* \leftarrow \emptyset$ , constants  $\alpha \geq 1, \beta \geq 1$  which depend on the constant  $\epsilon$ .
  - 3: Divide indices  $\{1, \dots, n\}$  into  $(\alpha, \beta)$  windows as prescribed by Definition 5.
  - 4: **for** window  $w = 1, \dots, k/\alpha$  **do**
  - 5:     **for** every slot  $s_j$  in window  $w, j = 1, \dots, \alpha\beta$  **do**
  - 6:         Concurrently for all subsequences of previous slots  $\tau \subseteq \{s_1, \dots, s_{j-1}\}$  of length  $|\tau| < \alpha$  in window  $w$ , call the online algorithm in Algorithm 1 with the following inputs:
    - number of items  $N = |s_j| + 1, \delta = \frac{\epsilon}{2}$ , and
    - item values  $I = (a_0, a_1, \dots, a_{N-1})$ , with
 
$$a_0 := \max_{x \in R_{1, \dots, w-1}} \Delta(x | S_{1, \dots, w-1} \cup \gamma(\tau))$$

$$a_\ell := \Delta(s_j(\ell) | S_{1, \dots, w-1} \cup \gamma(\tau)), \forall \ell = 1, \dots, N-1$$
 where  $s_j(\ell)$  denotes the  $\ell^{\text{th}}$  item in the slot  $s_j$ .
  - 7:         Let  $A_j(\tau)$  be the shortlist returned by Algorithm 1 for slot  $j$  and subsequence  $\tau$ .  
 Add all items except the dummy item 0 to the shortlist  $A$ . Let's  $A(j) = \bigcup_{\tau} A_j(\tau)$ .  
 That is,
 
$$A \leftarrow A \cup (A(j) \cap s_j)$$
  - 8:     **end for**
  - 9:     After seeing all items in window  $w$ , compute  $R_w, S_w$  as defined in (3) and (4) respectively.
  - 10:      $A^* \leftarrow A^* \cup (S_w \cap A)$
  - 11: **end for**
  - 12: return  $A, A^*$ .
- 

► **Definition 5** ( $(\alpha, \beta)$  windows). Let  $X_1, \dots, X_{k\beta}$  be a  $(n, k\beta)$ -ball-bin random set. Divide the indices  $\{1, \dots, n\}$  into  $k\beta$  slots, where the  $j$ -th slot,  $s_j$ , consists of  $X_j$  consecutive indices in the natural way, that is, slot 1 contains the first  $X_1$  indices, slot 2 contains the next  $X_2$  indices, etc. Next, we define  $k/\alpha$  windows, where window  $w$  consists of  $\alpha\beta$  consecutive slots, in the same manner as we assigned slots.

Thus, the  $q^{\text{th}}$  slot is composed of indices  $\{\ell, \dots, r\}$ , where  $\ell = X_1 + \dots + X_{q-1} + 1$  and  $r = X_1 + \dots + X_q$ . Further, if the ordered the input is  $\bar{a}_1, \dots, \bar{a}_n$ , then we say that the items inside the slot  $s_q$  are  $\bar{a}_\ell, \bar{a}_{\ell+1}, \dots, \bar{a}_r$ . To reduce notation, when clear from context, we will use  $s_q$  and  $w$  to also indicate the *set of items* in the slot  $s_q$  and window  $w$  respectively.

When  $\alpha$  and  $\beta$  are large enough constants, some useful properties can be obtained from the construction of these windows and slots. First, roughly  $\alpha$  items from the optimal set  $S^*$  are likely to lie in each of these windows; and further, it is unlikely that two items from  $S^*$  will appear in the same slot. (These statements will be made more precise in the analysis where precise setting of  $\alpha, \beta$  in terms of  $\epsilon$  will be provided.) Consequently, our algorithm can focus on identifying a constant number (roughly  $\alpha$ ) of optimal items from each of these windows, with at most one item coming from each of the  $\alpha\beta$  slots in a window. The core of our algorithm is a subroutine that accomplishes this task in an online manner using a shortlist of constant size in each window.



To implement this task, we use a greedy selection method that considers all possible  $\alpha$  sized subsequences of the  $\alpha\beta$  slots in a window, and aims to identify the subsequence that maximizes the increment over the ‘best’ items identified so far. More precisely, for any subsequence  $\tau = (s_1, \dots, s_\ell)$  of the  $\alpha\beta$  slots in window  $w$ , we define a ‘greedy’ subsequence  $\gamma(\tau)$  of items as:

$$\gamma(\tau) := \{i_1, \dots, i_\ell\} \quad (1)$$

where

$$i_j := \arg \max_{i \in s_j \cup R_{1, \dots, w-1}} f(S_{1, \dots, w-1} \cup \{i_1, \dots, i_{j-1}\} \cup \{i\}) - f(S_{1, \dots, w-1} \cup \{i_1, \dots, i_{j-1}\}). \quad (2)$$

In (2) and in the rest of the paper, we use shorthand  $S_{1, \dots, w}$  to denote  $S_1 \cup \dots \cup S_w$ , and  $R_{1, \dots, w}$  to denote  $R_1 \cup \dots \cup R_w$ , etc. We also will take unions of subsequences, which we interpret as the union of the elements in the subsequences. Here  $R_w$  is defined to be the union of all greedy subsequences of length  $\alpha$ , and  $S_w$  to be the best subsequence among those. That is,

$$R_w = \cup_{\tau: |\tau|=\alpha} \gamma(\tau) \quad (3)$$

and

$$S_w = \gamma(\tau^*), \quad (4)$$

where

$$\tau^* := \arg \max_{\tau: |\tau|=\alpha} f(S_{1, \dots, w-1} \cup \gamma(\tau)) - f(S_{1, \dots, w-1}). \quad (5)$$

Note that  $i_j$  (refer to (2)) can be set as either an item in slot  $s_j$  or an item *from a previous greedy subsequence* in  $R_1 \cup \dots \cup R_{w-1}$ . The significance of the latter relaxation will become clear in the analysis.

As such, identifying the sets  $R_w$  and  $S_w$  involves looking forward in a slot  $s_j$  to find the best item (according to the given criterion in (2)) among all the items in the slot. To obtain an online implementation of this procedure, we use an online subroutine that employs the algorithm (Algorithm 1) for the basic secretary problem with shortlists described earlier. This online procedure will result in selection of a set  $H_w$  potentially larger than  $R_w$ , while ensuring that each element from  $R_w$  is part of  $H_w$  with a high probability  $1 - \delta$  at the cost of adding extra  $\log(1/\delta)$  items to the shortlist. Note that  $R_w$  and  $S_w$  can be computed *exactly* at the end of window  $w$ .

Algorithm 2 summarizes the overall structure of our algorithm. In the algorithm, for any item  $i$  and set  $V$ , we define  $\Delta_f(i|V) := f(V \cup \{i\}) - f(V)$ .

The algorithm returns both the shortlist  $A$  which we show to be of size  $O(k)$  in the following proposition, as well as a set  $A^* = \cup_w (S_w \cap A)$  of size at most  $k$  to compete with  $S^*$ . In the next section, we will show that  $\mathbb{E}[f(A^*)] \geq (1 - \frac{1}{e} - \epsilon - O(\frac{1}{k}))f(S^*)$  to provide a bound on the competitive ratio of this algorithm.

► **Proposition 6.** *Given  $k, n$ , and any constant  $\alpha, \beta$  and  $\epsilon$ , the size of shortlist  $A$  selected by Algorithm 2 is of size at most  $4k\beta \binom{\alpha\beta}{\alpha} \log(2/\epsilon) = O(k)$ .*

**Proof.** For each window  $w = 1, \dots, k/\alpha$ , and for each of the  $\alpha\beta$  slots in this window, lines 6 through 7 in Algorithm 2 runs Algorithm 1 for  $\binom{\alpha\beta}{\alpha}$  times (for all  $\alpha$  length subsequences). By construction of Algorithm 1, for each run it will add at most  $L \leq 4 \log(2/\epsilon)$  items to the shortlist. Therefore, over all windows, Algorithm 2 adds at most  $\frac{k}{\alpha} \times \alpha\beta \binom{\alpha\beta}{\alpha} L = O(k)$  items to the shortlist. ◀

### 3 Bounding the competitive ratio (Proof of Theorem 1)

In this section we show that for any  $\epsilon \in (0, 1)$ , Algorithm 2 with an appropriate choice of constants  $\alpha, \beta$ , achieves the competitive ratio claimed in Theorem 1 for the submodular  $k$ -secretary problem with shortlists.

Recall the following notation defined in the previous section. For any collection of sets  $V_1, \dots, V_\ell$ , we use  $V_{1, \dots, \ell}$  to denote  $V_1 \cup \dots \cup V_\ell$ . Also, recall that for any item  $i$  and set  $V$ , we denote  $\Delta_f(i|V) := f(V \cup \{i\}) - f(V)$ .

#### Proof overview

The proof is divided into two parts. We first show a lower bound on the ratio  $\mathbb{E}[f(\cup_w S_w)]/\text{OPT}$  in Proposition 24, where  $S_w$  is the subset of items as defined in (4) for every window  $w$ . Later in Proposition 27, we use the said bound to derive a lower bound on the ratio  $\mathbb{E}[f(A^*)]/\text{OPT}$ , where  $A^* = A \cap (\cup_w S_w)$  is the subset of the shortlist returned by Algorithm 2.

Specifically, in Proposition 24, we provide settings of parameters  $\alpha, \beta$  such that  $\mathbb{E}[f(\cup_w S_w)] \geq (1 - \frac{1}{e} - \frac{\epsilon}{2} - O(\frac{1}{k})) \text{OPT}$ . A central idea in the proof of this result is to show that for every window  $w$ , given  $R_{1, \dots, w-1}$ , the items tracked from the previous windows, any of the  $k$  items from the optimal set  $S^*$  has at least  $\frac{\alpha}{k}$  probability to appear either in window  $w$ , or among the tracked items  $R_{1, \dots, w-1}$ . Further, the items from  $S^*$  that appear in window  $w$ , appear independently, and in a uniformly random slot in this window. (See Lemma 15.) These observations allow us to show that, in each window  $w$ , there exists a subsequence  $\tilde{\tau}_w$  of close to  $\alpha$  slots, such that the greedy sequence of items  $\gamma(\tilde{\tau}_w)$  will be almost “as good as” a randomly chosen sequence of  $\alpha$  items from  $S^*$ . More precisely, denoting  $\gamma(\tilde{\tau}_w) = (i_1, \dots, i_t)$ , in Lemma 19, for all  $j = 1, \dots, t$ , we lower bound the increment in function value  $f(\dots)$  on adding  $i_j$  over the items in  $S_{1, \dots, w-1} \cup i_{1, \dots, j-1}$  as:

$$\begin{aligned} & \mathbb{E}[\Delta_f(i_j | S_{1, \dots, w-1} \cup \{i_1, \dots, i_{j-1}\}) | T_{1, \dots, w-1}, i_1, \dots, i_{j-1}] \\ & \geq \frac{1}{k} \left( \left(1 - \frac{\alpha}{k}\right) f(S^*) - f(S_{1, \dots, w-1} \cup \{i_1, \dots, i_{j-1}\}) \right). \end{aligned}$$

We then deduce (using standard techniques for the analysis of greedy algorithm for submodular functions) that

$$\begin{aligned} & \mathbb{E} \left[ \left(1 - \frac{\alpha}{k}\right) f(S^*) - f(S_{1, \dots, w-1} \cup \gamma(\tilde{\tau}_w)) \mid S_{1, \dots, w-1} \right] \\ & \leq e^{-t/k} \left( \left(1 - \frac{\alpha}{k}\right) f(S^*) - f(S_{1, \dots, w-1}) \right). \end{aligned}$$

Now, since the length  $t$  of  $\tilde{\tau}_w$  is close to  $\alpha$  (as we show in Lemma 21) and since  $S_w = \gamma(\tau^*)$  with  $\tau^*$  defined as the “best” subsequence of length  $\alpha$  (refer to definition of  $\tau^*$  in (5)), we can show that a similar inequality holds for  $S_w = \gamma(\tau^*)$ , i.e.,

$$\begin{aligned} & \left(1 - \frac{\alpha}{k}\right) f(S^*) - \mathbb{E}[f(S_{1, \dots, w-1} \cup S_w) \mid S_{1, \dots, w-1}] \\ & \leq e^{-\alpha/k} (1 - \delta') \left( \left(1 - \frac{\alpha}{k}\right) f(S^*) - f(S_{1, \dots, w-1}) \right), \end{aligned}$$

where  $\delta' \in (0, 1)$  depends on the setting of  $\alpha, \beta$ . (See Lemma 23.) Then repeatedly applying this inequality for  $w = 1, \dots, k/\alpha$ , and setting  $\delta, \alpha, \beta$  appropriately in terms of  $\epsilon$ , we can obtain  $\mathbb{E}[f(S_{1, \dots, w})] \geq (1 - \frac{1}{e} - \frac{\epsilon}{2} - \frac{1}{k}) f(S^*)$ , completing the proof of Proposition 24.

However, a remaining difficulty is that while the algorithm keeps a track of the set  $S_w$  for every window  $w$ , it may not have been able to add all the items in  $S_w$  to the shortlist  $A$  during the online processing of the inputs in that window. In the proof of Proposition 27, we show that in fact the

algorithm will add most of the items in  $\cup_w S_w$  to the shortlist. More precisely, we show that given that an item  $i$  is in  $S_w$ , it will be in shortlist  $A$  with probability  $1 - \delta$ , where  $\delta$  is the parameter used while calling Algorithm 1 in Algorithm 2. Therefore, using properties of submodular functions it follows that with  $\delta = \epsilon/2$ ,  $\mathbb{E}[f(A^*)] = \mathbb{E}[f(\cup_w S_w \cap A)] \geq (1 - \frac{\epsilon}{2})\mathbb{E}[f(\cup_w S_w)]$  (see Proposition 27). Combining this with the lower bound  $\frac{\mathbb{E}[f(\cup_w S_w)]}{\text{OPT}} \geq (1 - \frac{1}{e} - \frac{\epsilon}{2} - O(\frac{1}{k}))$  proven in Proposition 24, we complete the proof of competitive ratio bound stated in Theorem 1.

### 3.1 Preliminaries

The following properties of submodular functions are well known (e.g., see [8, 12, 14]).

► **Lemma 7.** *Given a monotone submodular function  $f$ , and subsets  $A, B$  in the domain of  $f$ , we use  $\Delta_f(A|B)$  to denote  $f(A \cup B) - f(B)$ . For any set  $A$  and  $B$ ,  $\Delta_f(A|B) \leq \sum_{a \in A \setminus B} \Delta_f(a|B)$ .*

► **Lemma 8.** *Denote by  $A(p)$  a random subset of  $A$  where each element has a probability at least  $p$  to appear in  $A$  (not necessarily independently). Then  $\mathbb{E}[f(A(p))] \geq (1 - p)f(\emptyset) + (p)f(A)$ .*

We will use the following well known deviation inequality for martingales (or supermartingales/submartingales).

► **Lemma 9 (Azuma-Hoeffding inequality).** *Suppose  $\{X_k : k = 0, 1, 2, 3, \dots\}$  is a martingale (or super-martingale) and  $|X_k - X_{k-1}| < c_k$ , almost surely. Then for all positive integers  $N$  and all positive reals  $r$ ,*

$$P(X_N - X_0 \geq r) \leq \exp\left(\frac{-r^2}{2 \sum_{k=1}^N c_k^2}\right).$$

And symmetrically (when  $X_k$  is a sub-martingale):

$$P(X_N - X_0 \leq -r) \leq \exp\left(\frac{-r^2}{2 \sum_{k=1}^N c_k^2}\right).$$

► **Lemma 10 (Chernoff bound for Bernoulli r.v.).** *Let  $X = \sum_{i=1}^N X_i$ , where  $X_i = 1$  with probability  $p_i$  and  $X_i = 0$  with probability  $1 - p_i$ , and all  $X_i$  are independent. Let  $\mu = \mathbb{E}(X) = \sum_{i=1}^N p_i$ . Then,*

$$P(X \geq (1 + \delta)\mu) \leq e^{-\delta^2 \mu / (2 + \delta)}$$

for all  $\delta > 0$ , and

$$P(X \leq (1 - \delta)\mu) \leq e^{-\delta^2 \mu / 2}$$

for all  $\delta \in (0, 1)$ .

### 3.2 Some useful properties of $(\alpha, \beta)$ windows

All the proofs in this section are omitted and are provided in the full version [1].

We first prove some useful properties of  $(\alpha, \beta)$  windows defined in Definition 5 and used in Algorithm 2. The first observation is that every item will appear uniformly at random in one of the  $k\beta$  slots in  $(\alpha, \beta)$  windows.

► **Definition 11.** For each item  $e \in \bar{I}$ , define  $Y_e \in [k\beta]$  as the random variable indicating the slot in which  $e$  appears. We call vector  $Y \in [k\beta]^n$  a *configuration*.

► **Lemma 12.** *Random variables  $\{Y_e\}_{e \in \bar{I}}$  are i.i.d. with uniform distribution on all  $k\beta$  slots.*

This follows from the uniform random order of arrivals, and the use of the balls in bins process to determine the number of items in a slot during the construction of  $(\alpha, \beta)$  windows.

Next, we make some observations about the probability of assignment of items in  $S^*$  to the slots in a window  $w$ , given the sets  $R_{1, \dots, w-1}, S_{1, \dots, w-1}$  (refer to (3), (4) for definition of these sets). To aid analysis, we define the following new random variable  $T_w$  that will track all the useful information from a window  $w$ .

► **Definition 13.** Define  $T_w := \{(\tau, \gamma(\tau))\}_\tau$ , for all  $\alpha$ -length subsequences  $\tau = (s_1, \dots, s_\alpha)$  of the  $\alpha\beta$  slots in window  $w$ . Here,  $\gamma(\tau)$  is a sequence of items as defined in (1). Also define  $\text{Supp}(T_{1, \dots, w}) := \{e | e \in \gamma(\tau) \text{ for some } (\tau, \gamma(\tau)) \in T_{1, \dots, w}\}$  (Note that  $\text{Supp}(T_{1, \dots, w}) = R_{1, \dots, w}$ ).

► **Lemma 14.** For any window  $w \in [W]$ ,  $T_{1, \dots, w}$  and  $S_{1, \dots, w}$  are independent of the ordering of elements within any slot, and are determined by the configuration  $Y$ .

Following the above lemma, given a configuration  $Y$ , we will some times use the notation  $T_{1, \dots, w}(Y)$  and  $S_{1, \dots, w}(Y)$  to make this mapping explicit.

► **Lemma 15.** For any item  $i \in S^*$ , window  $w \in \{1, \dots, W\}$ , and slot  $s$  in window  $w$ , define

$$p_{is} := \Pr(i \in s \cup \text{Supp}(T) | T_{1, \dots, w-1} = T). \quad (6)$$

Then, for any pair of slots  $s', s''$  in windows  $w, w+1, \dots, W$ ,

$$p_{is'} = p_{is''} \geq \frac{1}{k\beta}. \quad (7)$$

► **Lemma 16.** For any window  $w$ ,  $i, j \in S^*$ ,  $i \neq j$  and  $s, s' \in w$ , the random variables  $\mathbf{1}(Y_i = s | T_{1, \dots, w-1} = T)$  and  $\mathbf{1}(Y_j = s' | T_{1, \dots, w-1} = T)$  are independent. That is, given  $T_{1, \dots, w-1} = T$ , items  $i, j \in S^*$ ,  $i \neq j$  appear in any slot  $s$  in  $w$  independently.

### 3.3 Bounding $\mathbb{E}[f(\cup_w S_w)]/\text{OPT}$

In this section, we use the observations from the previous sections to show the existence of a random subsequence of slots  $\tilde{\tau}_w$  of window  $w$  such that we can lower bound  $f(S_{1, \dots, w-1} \cup \gamma(\tilde{\tau}_w)) - f(S_{1, \dots, w-1})$  in terms of  $\text{OPT} - f(S_{1, \dots, w-1})$ . This will be used to lower bound increment  $\Delta_f(S_w | S_{1, \dots, w-1}) = f(S_{1, \dots, w-1} \cup \gamma(\tau^*)) - f(S_{1, \dots, w-1})$  in every window.

► **Definition 17** ( $Z_s$  and  $\tilde{\gamma}_w$ ). Create sets of items  $Z_s, \forall s \in w$  as follows: for every slot  $s$ , add every item from  $i \in S^* \cap s$  independently with probability  $\frac{1}{k\beta p_{is}}$  to  $Z_s$ . Then, for every item  $i \in S^* \cap \text{Supp}(T)$ , with probability  $\alpha/k$ , add  $i$  to  $Z_s$  for a randomly chosen slot  $s$  in  $w$ . Define subsequence  $\tilde{\tau}_w$  as the sequence of slots with  $Z_s \neq \emptyset$ .

► **Lemma 18.** Given any  $T_{1, \dots, w-1} = T$ , for any slot  $s$  in window  $w$ , all  $i, i' \in S^*$ ,  $i \neq i'$  will appear in  $Z_s$  independently with probability  $\frac{1}{k\beta}$ . Also, given  $T$ , for every  $i \in S^*$ , the probability to appear in  $Z_s$  is equal for all slots  $s$  in window  $w$ . Further, each  $i \in S^*$  occurs in  $Z_s$  for at most one slot  $s$ .

**Proof.** First consider  $i \in S^* \cap \text{Supp}(T)$ . Then,  $\Pr(i \in Z_s | T) = \frac{\alpha}{k} \times \frac{1}{\alpha\beta} = \frac{1}{k\beta}$  by construction. Also, the event  $i \in Z_s | T$  is independent from  $i' \in Z_s | T$  for any  $i' \in S^*$  as  $i$  and  $i'$  are independently assigned to a  $Z_s$  in construction. Further, items in  $S^* \cap \text{Supp}(T)$  are assigned with equal probability to slots in window  $w$ .

Now, consider  $i \in S^*$ ,  $i \notin \text{Supp}(T)$ . Then, for all slots  $s$  in window  $w$ ,

$$\Pr(i \in Z_s | T) = \Pr(Y_i = s | T) \frac{1}{p_{is} k\beta} = p_{is} \times \frac{1}{p_{is} k\beta} = \frac{1}{k\beta},$$

where  $p_{is}$  is defined in (6). We used that  $p_{is} = \Pr(Y_i = s | T)$  for  $i \notin \text{Supp}(T)$ . Independence of events  $i \in Z_s | T$  for items in  $S^* \setminus \text{Supp}(T)$  follows from Lemma 16, which ensures  $Y_i = s | T$  and  $Y_j = s' | T$  are independent for  $i \neq j$ ; and from independent selection among items with  $Y_i = s$  into  $Z_s$ .

The fact that every  $i \in S^*$  occurs in at most one  $Z_s$  follows from construction:  $i$  is assigned to  $Z_s$  of only one slot if  $i \in \text{Supp}(T)$ ; and for  $i \notin \text{Supp}(T)$ , it can only appear in  $Z_s$  if  $i$  appears in slot  $s$ . ◀

► **Lemma 19.** Given the sequence  $\tilde{\tau}_w = (s_1, \dots, s_t)$  defined in Definition 17, let  $\gamma(\tilde{\tau}_s) = (i_1, \dots, i_t)$ , with  $\gamma(\cdot)$  as defined in (1). Then, for all  $j = 1, \dots, t$ ,

$$\begin{aligned} & \mathbb{E}[\Delta_f(i_j | S_{1, \dots, w-1} \cup \{i_1, \dots, i_{j-1}\}) | T_{1, \dots, w-1}, i_1, \dots, i_{j-1}] \\ & \geq \frac{1}{k} \left( \left(1 - \frac{\alpha}{k}\right) f(S^*) - f(S_{1, \dots, w-1} \cup \{i_1, \dots, i_{j-1}\}) \right). \end{aligned}$$

**Proof.** For any slot  $s'$  in window  $w$ , let  $\{s : s \succ_w s'\}$  denote all the slots that appear after  $s'$  in the sequence of slots in window  $w$ .

Now, using Lemma 18, for any slot  $s$  such that  $s \succ_w s_{j-1}$ , we have that the random variables  $\mathbf{1}(i \in Z_s | Z_{s_1} \cup \dots \cup Z_{s_{j-1}})$  are i.i.d. for all  $i \in S^* \setminus \{Z_{s_1} \cup \dots \cup Z_{s_{j-1}}\}$ . Next, we show that the probabilities  $\Pr(i \in Z_{s_j} | Z_{s_1} \cup \dots \cup Z_{s_{j-1}})$  are identical for all  $i \in S^* \setminus \{Z_{s_1} \cup \dots \cup Z_{s_{j-1}}\}$ :

$$\begin{aligned} & \Pr(i \in Z_{s_j} | Z_{s_1} \cup \dots \cup Z_{s_{j-1}}) \\ &= \sum_{s: s \succ_w s_{j-1}} \Pr(i \in Z_s, s = s_j | Z_{s_1} \cup \dots \cup Z_{s_{j-1}}) \\ &= \sum_{s: s \succ_w s_{j-1}} \Pr(i \in Z_s | s = s_j, Z_{s_1} \cup \dots \cup Z_{s_{j-1}}) \Pr(s = s_j | Z_{s_1} \cup \dots \cup Z_{s_{j-1}}). \end{aligned}$$

Now, from Lemma 18, the probability  $\Pr(i \in Z_s | s = s_j, Z_{s_1} \cup \dots \cup Z_{s_{j-1}})$  must be identical for all  $i \notin Z_{s_1} \cup \dots \cup Z_{s_{j-1}}$ . Therefore, from above we have that for all  $i, i' \in S^* \setminus \{Z_{s_1} \cup \dots \cup Z_{s_{j-1}}\}$ ,

$$\Pr(i \in Z_{s_j} | Z_{s_1} \cup \dots \cup Z_{s_{j-1}}) = \Pr(i' \in Z_{s_j} | Z_{s_1} \cup \dots \cup Z_{s_{j-1}}) \geq \frac{1}{k}. \quad (8)$$

The lower bound of  $1/k$  followed from the fact that at least one of the items from  $S^* \setminus \{Z_{s_1} \cup \dots \cup Z_{s_{j-1}}\}$  must appear in  $Z_{s_j}$  for  $s_j$  to be included in  $\tilde{\tau}_w$ . Thus, each of these probabilities is at least  $1/k$ . In other words, if an item is randomly picked from  $Z_{s_j}$ , it will be  $i$  with probability at least  $1/k$ , for all  $i \in S^* \setminus \{Z_{s_1} \cup \dots \cup Z_{s_{j-1}}\}$ .

Now, by definition of  $\gamma(\cdot)$  (refer to (1)),  $i_j$  is chosen greedily to maximize the increment  $\Delta_f(i | S_{1, \dots, w-1} \cup i_{1, \dots, s-1})$  over all  $i \in s_j \cup \text{Supp}(T_{1, \dots, w-1}) \supseteq Z_{s_j}$ . Therefore, we can lower bound the increment provided by  $i_j$  by that provided by a randomly picked item from  $Z_{s_j}$ . By using monotonicity of  $f$ ,

$$\begin{aligned} & \mathbb{E}[\Delta_f(i_j | S_{1, \dots, w-1} \cup \{i_1, \dots, i_{j-1}\}) | T_{1, \dots, w-1} = T, i_1, \dots, i_{j-1}] \\ \text{(by (8))} & \geq \frac{1}{k} \mathbb{E} \left[ \sum_{i \in S^* \setminus \{Z_1, \dots, Z_{s_{j-1}}\}} \mathbb{E}[\Delta_f(i | S_{1, \dots, w-1} \cup \{i_1, \dots, i_{j-1}\}) | T, i_1, \dots, i_{j-1}] \right] \\ \text{(by Lemma 7)} & \geq \frac{1}{k} \mathbb{E} [ (f(S^* \setminus \{Z_1, \dots, Z_{s_{j-1}}\}) - f(S_{1, \dots, w-1} \cup \{i_1, \dots, i_{j-1}\})) | T ] \\ & \geq \frac{1}{k} \mathbb{E} [ (f(S^* \setminus \cup_{s' \in w} Z_{s'}) - f(S_{1, \dots, w-1} \cup \{i_1, \dots, i_{j-1}\})) | T ] \\ \text{(by Lemma 18 and 8)} & \geq \frac{1}{k} \left( \left(1 - \frac{\alpha}{k}\right) f(S^*) - f(S_{1, \dots, w-1} \cup \{i_1, \dots, i_{j-1}\}) \right) \end{aligned}$$

The last inequality uses the observation from Lemma 18 that given  $T$ , every  $i \in S^*$  appears in  $\cup_{s' \in w} Z_{s'}$  independently with probability  $\alpha/k$ , so that every  $i \in S^*$  appears in  $S^* \setminus \cup_{s' \in w} Z_{s'}$  independently with probability  $1 - \frac{\alpha}{k}$ ; along with Lemma 8 for submodular function  $f$ .  $\blacktriangleleft$

Using standard techniques for the analysis of greedy algorithm, the following corollary of the previous lemma can be derived: given any  $T_{1, \dots, w-1} = T$ :

**► Lemma 20.**

$$\mathbb{E} \left[ \left(1 - \frac{\alpha}{k}\right) f(S^*) - f(S_{1, \dots, w-1} \cup \gamma(\tilde{\tau}_w)) | T \right] \leq \mathbb{E} \left[ e^{-\frac{|\tilde{\tau}_w|}{k}} | T \right] \left( \left(1 - \frac{\alpha}{k}\right) f(S^*) - f(S_{1, \dots, w-1}) \right).$$

**Proof.** Let  $\pi_0 = (1 - \frac{\alpha}{k})f(S^*) - \mathbb{E}[f(S_{1, \dots, w-1}) | T_{1, \dots, w-1} = T]$ , and for  $j \geq 1$ ,

$$\pi_j := \left(1 - \frac{\alpha}{k}\right) f(S^*) - \mathbb{E}[f(S_{1, \dots, w-1} \cup \{i_1, \dots, i_j\}) | T_{1, \dots, w-1} = T, i_1, \dots, i_{j-1}],$$

Then, subtracting and adding  $(1 - \frac{\alpha}{k})f(S^*)$  from the left hand side of the previous lemma, and taking expectation conditional on  $T_{1, \dots, w-1} = T, i_1, \dots, i_{j-2}$ , we get

$$-\mathbb{E}[\pi_j | T, i_1, \dots, i_{j-2}] + \pi_{j-1} \geq \frac{1}{k} \pi_{j-1}$$

## 1:14 Submodular Secretary Problem with Shortlists

which implies

$$\mathbb{E}[\pi_j | T, i_1, \dots, i_{j-2}] \leq \left(1 - \frac{1}{k}\right) \pi_{j-1} \leq \left(1 - \frac{1}{k}\right)^j \pi_0 .$$

By the martingale stopping theorem, this implies:

$$\mathbb{E}[\pi_t | T] \leq \mathbb{E} \left[ \left(1 - \frac{1}{k}\right)^t | T \right] \pi_0 \leq \mathbb{E} [e^{-t/k} | T] \pi_0$$

where stopping time  $t = |\tilde{\tau}_w|$ . ( $t = |\tilde{\tau}_w| \leq \alpha\beta$  is bounded, therefore, the martingale stopping theorem can be applied). ◀

Next, we compare  $\gamma(\tilde{\tau}_w)$  to  $S_w = \gamma(\tau^*)$ . Here,  $\tau^*$  was defined as the ‘best’ greedy subsequence of length  $\alpha$  (refer to (4) and (5)). To compare it with  $\tilde{\tau}_w$ , we need a bound on size of  $\tilde{\tau}_w$ .

► **Lemma 21.** *For any real  $\delta \in (0, 1)$ , and if  $k \geq \alpha\beta$ ,  $\alpha \geq 8 \log(\beta)$  and  $\beta \geq 8$ , then given any  $T_{1, \dots, w-1} = T$ ,*

$$(1 - \delta) \left(1 - \frac{4}{\beta}\right) \alpha \leq |\tilde{\tau}_w| \leq (1 + \delta) \alpha,$$

with probability at least  $1 - \exp(-\frac{\delta^2 \alpha}{8\beta})$ .

**Proof.** Appears in the full version. ◀

► **Lemma 22 (Corollary of Lemma 21).** *For any real  $\delta' \in (0, 1)$ , if parameters  $k, \alpha, \beta$  satisfy  $k \geq \alpha\beta$ ,  $\beta \geq \frac{8}{(\delta')^2}$ ,  $\alpha \geq 8\beta^2 \log(1/\delta')$ , then given any  $T_{1, \dots, w-1} = T$ , with probability at least  $1 - \delta' e^{-\alpha/k}$ ,*

$$|\tilde{\tau}_w| \geq (1 - \delta') \alpha .$$

► **Lemma 23.** *For any real  $\delta' \in (0, 1)$ , if parameters  $k, \alpha, \beta$  satisfy  $k \geq \alpha\beta$ ,  $\beta \geq \frac{8}{(\delta')^2}$ ,  $\alpha \geq 8\beta^2 \log(1/\delta')$ , then*

$$\mathbb{E} \left[ \frac{k - \alpha}{k} OPT - f(S_{1, \dots, w}) | T_{1, \dots, w-1} \right] \leq (1 - \delta') e^{-\alpha/k} \left( \frac{k - \alpha}{k} OPT - f(S_{1, \dots, w-1}) \right) .$$

**Proof.** The lemma follows from substituting Lemma 22 in Lemma 20. ◀

Now, we can deduce the following proposition.

► **Proposition 24.** *For any real  $\delta' \in (0, 1)$ , if parameters  $k, \alpha, \beta$  satisfy  $k \geq \alpha\beta$ ,  $\beta \geq \frac{8}{(\delta')^2}$ ,  $\alpha \geq 8\beta^2 \log(1/\delta')$ , then the set  $S_{1, \dots, W}$  tracked by Algorithm 2 satisfies*

$$\mathbb{E}[f(S_{1, \dots, W})] \geq (1 - \delta')^2 (1 - 1/e) OPT.$$

**Proof.** By multiplying the inequality Lemma 23 from  $w = 1, \dots, W$ , where  $W = k/\alpha$ , we get

$$\mathbb{E}[f(S_{1, \dots, W})] \geq (1 - \delta') (1 - 1/e) \left(1 - \frac{\alpha}{k}\right) OPT.$$

Then, using  $1 - \frac{\alpha}{k} \geq 1 - \delta'$  because  $k \geq \alpha\beta \geq \frac{\alpha}{\delta'}$ , we obtain the desired statement. ◀

### 3.4 Bounding $\mathbb{E}[f(A^*)]/OPT$

Here, we compare  $f(S_{1, \dots, W})$  to  $f(A^*)$ , where  $A^* = S_{1, \dots, W} \cap A$ , with  $A$  being the shortlist returned by Algorithm 2. The main difference between the two sets is that in construction of shortlist  $A$ , Algorithm 1 is being used to compute the argmax in the definition of  $\gamma(\tau)$ , in an online manner. This argmax may not be computed exactly, so that some items from  $S_{1, \dots, W}$  may not be part of the shortlist  $A$ . We use the following guarantee for Algorithm 1 to bound the probability of this event.

► **Proposition 25.** *For any  $\delta \in (0, 1)$ , and input  $I = (a_1, \dots, a_N)$ , Algorithm 1 returns  $A^* = \max(a_1, \dots, a_N)$  with probability  $(1 - \delta)$ .*

The proof of the above proposition appears in the full version. Intuitively, it follows from the observation that if we select every item that improves the maximum of items seen so far, we would have selected  $\log(N)$  items in expectation. The exact proof involves showing that on waiting  $n\delta/2$  steps and then selecting maximum of every item that improves the maximum of items seen so far, we miss the maximum item with at most  $\delta$  probability, and select at most  $O(\log(1/\delta))$  items with probability  $1 - \delta$ .

► **Lemma 26.** *Let  $A$  be the shortlist returned by Algorithm 2, and  $\delta$  is the parameter used to call Algorithm 1 in Algorithm 2. Then, for given configuration  $Y$ , for any item  $a$  and window  $w$ , we have*

$$\Pr(a \in A | Y, a \in S_{1, \dots, w}) \geq 1 - \delta.$$

**Proof.** From Lemma 14 by conditioning on  $Y$ , the set  $S_{1, \dots, w}$  is determined. Now if  $a \in S_{1, \dots, w}$ , then for some slot  $s_j$  in an  $\alpha$  length subsequence  $\tau$  of some window  $w$ , we must have

$$a = \arg \max_{i \in s_j \cup R_{1, \dots, w-1}} f(S_{1, \dots, w-1} \cup \gamma(\tau) \cup \{i\}) - f(S_{1, \dots, w-1} \cup \gamma(\tau)).$$

Let  $w'$  be the first such window,  $\tau', s_{j'}$  be the corresponding subsequence and slot. Then, it must be true that

$$a = \arg \max_{i \in s_{j'}} f(S_{1, \dots, w'-1} \cup \gamma(\tau') \cup \{i\}) - f(S_{1, \dots, w'-1} \cup \gamma(\tau')).$$

(Note that the argmax in above is not defined on  $R_{1, \dots, w'-1}$ ). The configuration  $Y$  only determines the set of items in the items in slot  $s_{j'}$ , the items in  $s_{j'}$  are still randomly ordered (refer to Lemma 14). Therefore, from Proposition 25, with probability  $1 - \delta$ ,  $a$  will be added to the shortlist  $A_{j'}(\tau')$  by Algorithm 1. Thus  $a \in A \supseteq A_{j'}(\tau')$  with probability at least  $1 - \delta$ . ◀

► **Proposition 27.**

$$\mathbb{E}[f(A^*)] := \mathbb{E}[f(S_{1, \dots, w} \cap A)] \geq (1 - \frac{\epsilon}{2})\mathbb{E}[f(S_{1, \dots, w})]$$

where  $A^* := S_{1, \dots, w} \cap A$  is the size  $k$  subset of shortlist  $A$  returned by Algorithm 2.

**Proof.** From the previous lemma, given any configuration  $Y$ , we have that each item of  $S_{1, \dots, w}$  is in  $A$  with probability at least  $1 - \delta$ , where  $\delta = \epsilon/2$  in Algorithm 2. Therefore using Lemma 8, the expected value of  $f(S_{1, \dots, w} \cap A)$  is at least  $(1 - \delta)\mathbb{E}[f(S_{1, \dots, w})]$ . ◀

## Proof of Theorem 1

Now, we can show that Algorithm 2 provides the results claimed in Theorem 1 for appropriate settings of  $\alpha, \beta$  in terms of  $\epsilon$ . Specifically for  $\delta' = \epsilon/4$ , set  $\alpha, \beta$  as smallest integers satisfying  $\beta \geq \frac{8}{(\delta')^2}, \alpha \geq 8\beta^2 \log(1/\delta')$ . Then, using Proposition 24 and Proposition 27, for  $k \geq \alpha\beta$  we obtain:

$$\mathbb{E}[f(A^*)] \geq (1 - \frac{\epsilon}{2})(1 - \delta')^2(1 - 1/e)\text{OPT} \geq (1 - \epsilon)(1 - 1/e)\text{OPT}.$$

This implies a lower bound of  $1 - \epsilon - 1/e - \alpha\beta/k = 1 - \epsilon - 1/e - O(1/k)$  on the competitive ratio. The  $O(k)$  bound on the size of the shortlist was demonstrated in Proposition 6.

## 4 Streaming (Proof of Theorem 2)

In this section, we show that Algorithm 2 can be implemented in a way that it uses a memory buffer of size at most  $\eta(k) = O(k)$ ; and the number of objective function evaluations for each arriving item is  $O(1 + \frac{k^2}{n})$ . This will allow us to obtain Theorem 2 as a corollary of Theorem 1.

In the current description of Algorithm 2, there are several steps in which the algorithm potentially needs to store  $O(n)$  previously seen items in order to compute the relevant quantities. First, in Step 6, in order to be able to compute  $\gamma(\tau)$  for all less than  $\alpha$  length subsequences  $\tau$  of slots  $s_1, \dots, s_{j-1}$ , the algorithm should have stored all the items that arrived in the slots  $s_1, \dots, s_{j-1}$ . However, this



memory requirement can be reduced by a small modification of the algorithm, so that at the end of iteration  $j - 1$ , the algorithm has already computed  $\gamma(\tau)$  for all such  $\tau$ , and stored them to be used in iteration  $j$ . In fact, this can be implemented in a memory efficient manner, in the following way. For every subsequence  $\tau$  of slots  $s_1, \dots, s_{j-1}$  of length  $< \alpha$ , consider prefix  $\tau' = \tau \setminus s_{j-1}$ . Assume  $\gamma(\tau')$  is available from iteration  $j - 2$ . If  $\tau' = \tau$ , then  $\gamma(\tau) = \gamma(\tau')$ . Otherwise, in Step 6 of iteration  $j - 1$ , the algorithm must have considered the subsequence  $\tau'$  while going through all subsequences of length less than  $\alpha$  of slots  $s_1, \dots, s_{j-2}$ . Now, modify the implementation of Step 6 so that the algorithm also tracks the (true) maximum  $M_{j-1}(\tau')$  of  $a_0, a_1, \dots, a_N$  for each  $\tau'$ . Then,  $\gamma(\tau)$  can be obtained by extending  $\gamma(\tau')$  by  $M_{j-1}(\tau')$ , i.e.,  $\gamma(\tau) = \{\gamma(\tau'), M_{j-1}(\tau')\}$ . Thus, at the end of iteration  $j - 1$ ,  $\gamma(\tau)$  would have been computed for all subsequences  $\tau$  relevant for iteration  $j$ , and so on. In order to store these  $\gamma(\tau)$  for every subsequence  $\tau$  (of at most  $\alpha$  slots from  $\alpha\beta$  slots), we require a memory buffer of size at most  $\alpha^2 \binom{\alpha\beta}{\alpha} = O(1)$ .

Secondly, across windows and slots, the algorithm keeps track of  $R_w, S_w, w = 1, \dots, k/\alpha$  where  $W = k/\alpha$ . In the current description of Algorithm 2, these sets are computed after seeing all the items in window  $w$  in Step 9. Thus, all the items arriving in that window would be needed to be stored in order to compute them, requiring  $O(n)$  memory buffer. However, the alternate implementation discussed in the previous paragraph reduces this memory requirement to  $O(k)$  as well. Using the above implementation, at the end of iteration  $\alpha\beta$  for the last slot  $s_{\alpha\beta}$  in window  $w$ , we would have computed and stored  $\gamma(\tau)$  for all the subsequences  $\tau$  of length  $\alpha$  of slots  $s_1, \dots, s_{\alpha\beta}$ .  $R_w$  is simply defined as union of all items in  $\gamma(\tau)$  over all such  $\tau$  (refer to (3)). And,  $S_w = \gamma(\tau^*)$  for the best subsequence  $\tau^*$  among these subsequences (refer to (4)). Thus, computing  $R_w$  and  $S_w$  does not require any additional memory buffer. Storing  $R_w$  and  $S_w$  for all windows requires a buffer of size at most  $\sum_w |R_w| + |S_w| = \frac{k}{\alpha} \times \alpha \binom{\alpha\beta}{\alpha} + k = O(k)$ . Therefore, the total buffer required to implement Algorithm 2 is of size  $O(k)$ .

Finally, let's bound the number of objective function evaluations for each arriving item. Each arriving item is processed in Step 6, where objective function is evaluated twice for each subsequence to compute the corresponding  $a_i$ . Since there are at most  $\binom{\alpha\beta}{\alpha}$  subsequences  $\tau$  for which this quantity is computed, the total number of times this computation is performed is bounded by  $2 \binom{\alpha\beta}{\alpha} = O(1)$ . For each  $\tau$ , we also compute  $a_0$  in the beginning of the slot. Computing  $a_0$  for each  $\tau$  involves taking max over all items in  $R_{1, \dots, w-1}$ , and requires  $2|R_{1, \dots, w-1}| \leq 2k \binom{\alpha\beta}{\alpha}$  evaluations of the objective function. Due to this computation, in the worst-case, the update time for an item can be  $2k \binom{\alpha\beta}{\alpha}^2 + 2 \binom{\alpha\beta}{\alpha} = O(k)$ . However, since  $a_0$  is computed *once* in the beginning of the slot for each  $\tau$ , the total update time over all items is bounded by  $2k \binom{\alpha\beta}{\alpha}^2 \times k\beta + \binom{\alpha\beta}{\alpha} \times n = O(k^2 + n)$ . Therefore the amortized update time for each item is  $O(1 + \frac{k^2}{n})$ . This concludes the proof of Theorem 2.

## 5 Impossibility Result (Proof of Theorem 3)

In this section we provide an upper bound showing the following:

► **Theorem 3.** *No online algorithm (even with unlimited computational power) can achieve a competitive ratio better than  $7/8 + o(1)$  for the submodular  $k$ -secretary problem with shortlists, while using a shortlist of size  $\eta(k) = o(n)$ .*

In the following proof, for simplicity of notation, we prove the desired bound for submodular  $(k + 1)$ -secretary problem. For any given  $n, k$ , we construct a set of instances of the submodular  $(k + 1)$ -secretary problem with shortlists such that any online algorithm that uses a shortlist of size  $\eta(k + 1)$  will have competitive ratio of at most  $\frac{7}{8} + \frac{\eta(k+1)}{2n}$  on a randomly selected instance from this set.

First, we define a monotone submodular function  $f$  as follows. The ground set consists of  $\frac{n}{2k} + n - 1$  items. There are two types of items,  $C$  and  $D$ , with  $L := n/2k$  items of type  $C$  and  $n - 1$  items of type  $D$ . We define  $f(\phi) := 0$ ,  $f(\{c\}) := k$  for  $c \in C$ , and  $f(\{d\}) := 1$  for all  $d \in D$ . Also there is a collection of  $L$  disjoint sets  $T_\ell = \{c^\ell, d_1^\ell, \dots, d_k^\ell\}$ ,  $\ell = 1, 2, \dots, L$ , such that  $c^\ell \in C$  and  $d_j^\ell \in D$ . We define  $f(T_\ell) := 2k$  for all  $\ell = 1, \dots, L$ . Now, let

$$g(t) := k + \frac{k}{2} + \dots + \frac{k}{2^{i-1}} + \frac{(t - ik)}{2^i},$$



where  $i = \lfloor t/k \rfloor$ . It is easy to see that  $g$  is a monotone submodular function.

Now, define  $f$  on the remaining subsets of the ground set as follows. For all  $S$  with  $|S| \geq 1$ ,

- $|S \cap C| \geq 2 \implies f(S) := 2k + 1$
- $|S \cap C| = 0 \implies f(S) := 1 + g(|S| - 1)$
- $|S \cap C| = 1 \implies S \cap C = \{c^\ell\}$  for some  $\ell \in [L] \implies$

$$f(S) := \min\{2k + 1, k + \frac{1}{2}g(|S| - 1) + \frac{k'}{2^{i+1}}\},$$

where  $k' = |S \cap \{d_1^\ell, \dots, d_k^\ell\}|$ ,  $i = \lfloor (|S| - 1)/k \rfloor$ .

Observe that since  $g(k) = k$ , for any such subset  $S$  of size at most  $k + 1$ , we have  $f(S) \leq k + \frac{k}{2} + \frac{k}{2} = 2k$ .

► **Lemma 28.**  *$f$  is a monotone submodular function.*

Now, denote  $D^\ell := T^\ell \cap D = \{d_1^\ell, \dots, d_k^\ell\}$  for  $\ell = 1, 2, \dots, L$ . Also, let  $D' = D \setminus (\bigcup_{\ell=1}^L D^\ell)$ . Now define  $L$  input instances  $\{I_\ell\}_{\ell=1, \dots, L}$ , each of size  $n$ , as follows. For any arbitrary subset  $\bar{D} \subseteq D'$  of size  $n - Lk - 1$ , define  $I_\ell = \bigcup_{i=1, \dots, L} D^i \cup \bar{D} \cup \{c^\ell\}$ , for  $\ell = 1, \dots, L$ . Thus, for instance  $I_\ell$ , the the optimal  $k + 1$  subset is  $T^\ell$  with value  $f(T^\ell) = 2k$ .

Now consider any algorithm for the submodular secretary problem with shortlists and cardinality constraint  $k + 1$ . We denote by  $Alg$  the set of  $\eta(k + 1)$  items selected by the algorithm as part of the shortlist. Let  $\bar{I}$  denote an instance chosen uniformly at random from  $I_\ell, \ell = 1, \dots, L$ . Let  $\pi$  denote a random ordering of  $n$  items in  $\bar{I}$ . We denote by random variable  $(\bar{I}, \pi)$  the randomly ordered input instance to the algorithm. Also we denote by  $\bar{T}, \bar{D}$  and  $\bar{c}$ , the corresponding  $T^\ell, D^\ell$  and  $c^\ell$ .

Now we claim

► **Lemma 29.**  $\mathbb{E}_{(\bar{I}, \pi)}[|Alg \cap \bar{D}|] \leq k/2 + \eta(k + 1)/L$ .

**Proof.** Appears in the full version [1]. ◀

Now on input  $\bar{I}$ , if the algorithm doesn't select  $\bar{c}$  as part of shortlist  $Alg$ , then by definition of  $f$  for sets that do not contain any item of type  $C$ , we have

$$f(A^*) := \max_{S \subseteq Alg: |S| \leq k+1} f(S) \leq 1 + g(k) < k + 2.$$

Otherwise, if algorithm selects  $\bar{c}$ , then by definition of  $f$ ,

$$f(A^*) := \max_{S \subseteq Alg: |S| \leq k+1} f(S) \leq \max_{S \subseteq Alg \setminus (\bar{D} \cup \{\bar{c}\}): |S| \leq k - |Alg \cap \bar{D}|} f(S \cup \bar{D} \cup \{\bar{c}\}) = k + \frac{k}{2} + \frac{1}{2}|Alg \cap \bar{D}|,$$

and therefore by lemma 29

$$\mathbb{E}[f(A^*)] \leq k + \frac{k}{2} + \frac{k}{4} + \frac{\eta(k + 1)}{2L} = \frac{7k}{4} + \frac{k\eta(k + 1)}{n}.$$

Since the optimal is equal to  $\mathbb{E}[f(\bar{T})] = 2k$ , the competitive ratio is upper bounded by

$$\frac{7}{8} + \frac{\eta(k + 1)}{2n}.$$

This proves a competitive ratio upper bound of  $\frac{7}{8} + o(1)$  when  $\eta(k + 1) = o(n)$ , to complete the proof of Theorem 3.

## References

- 1 Shipra Agrawal, Mohammad Shadravan, and Cliff Stein. Submodular Secretary Problem with Shortlists, 2018. [arXiv:1809.05082](#).
- 2 Shipra Agrawal, Zizhuo Wang, and Yinyu Ye. A Dynamic Near-Optimal Algorithm for Online Linear Programming. *Operations Research*, 62(4):876–890, 2014.
- 3 Miklos Ajtai, Nimrod Megiddo, and Orli Waarts. Improved Algorithms and Analysis for Secretary Problems and Generalizations. *SIAM J. Discret. Math.*, 14(1):1–27, January 2001.
- 4 Moshe Babaioff, Nicole Immorlica, David Kempe, and Robert Kleinberg. Online Auctions and Generalized Secretary Problems. *SIGecom Exch.*, 7(2):7:1–7:11, June 2008.
- 5 Ashwinkumar Badanidiyuru, Baharan Mirzasoleiman, Amin Karbasi, and Andreas Krause. Streaming Submodular Maximization: Massive Data Summarization on the Fly. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '14, pages 671–680, New York, NY, USA, 2014. ACM.
- 6 Mohammadhossein Bateni, Mohammadtaghi Hajiaghayi, and Morteza Zadimoghaddam. Submodular Secretary Problem and Extensions. *ACM Trans. Algorithms*, 9(4):32:1–32:23, October 2013.
- 7 Niv Buchbinder, Moran Feldman, and Mohit Garg. Online Submodular Maximization: Beating 1/2 Made Simple. *arXiv preprint*, 2018. [arXiv:1807.05529](#).
- 8 Niv Buchbinder, Moran Feldman, Joseph (Seffi) Naor, and Roy Schwartz. Submodular Maximization with Cardinality Constraints. In *Proceedings of the Twenty-fifth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '14, pages 1433–1452. Society for Industrial and Applied Mathematics, 2014.
- 9 Amit Chakrabarti and Sagar Kale. Submodular maximization meets streaming: matchings, matroids, and more. *Mathematical Programming*, 154(1):225–247, December 2015.
- 10 Nikhil Devanur and Thomas Hayes. The Adwords Problem: Online Keyword Matching with Budgeted Bidders under Random Permutations. In *ACM EC*, 2009.
- 11 E. B. Dynkin. The optimum choice of the instant for stopping a Markov process. *Soviet Math. Dokl*, 4, 1963.
- 12 Uriel Feige, Vahab S. Mirrokni, and Jan Vondrák. Maximizing Non-monotone Submodular Functions. *SIAM J. Comput.*, 40(4):1133–1153, July 2011.
- 13 J. Feldman, M. Henzinger, N. Korula, V. Mirrokni, and C. Stein. Online stochastic packing applied to display ad allocation. *Algorithms-ESA 2010*, pages 182–194, 2010.
- 14 Moran Feldman and Rico Zenklusen. The Submodular Secretary Problem Goes Linear. In *Proceedings of the 2015 IEEE 56th Annual Symposium on Foundations of Computer Science (FOCS)*, FOCS '15, pages 486–505. IEEE Computer Society, 2015.
- 15 Thomas S Ferguson et al. Who solved the secretary problem? *Statistical science*, 4(3):282–289, 1989.
- 16 Anupam Gupta, Aaron Roth, Grant Schoenebeck, and Kunal Talwar. Constrained Non-monotone Submodular Maximization: Offline and Secretary Algorithms. In *Proceedings of the 6th International Conference on Internet and Network Economics*, WINE'10, pages 246–257. Springer-Verlag, 2010.
- 17 Tom Hess and Sivan Sabato. The submodular secretary problem under a cardinality constraint and with limited resources. *CoRR*, abs/1702.03989, 2017. [arXiv:1702.03989](#).
- 18 Bala Kalyanasundaram and Kirk Pruhs. Speed is as powerful as clairvoyance. *J. ACM*, 47(4):617–643, 2000.

- 19 Michael Kapralov, Ian Post, and Jan Vondrák. Online Submodular Welfare Maximization: Greedy is Optimal. In *Proceedings of the Twenty-fourth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '13, pages 1216–1225. Society for Industrial and Applied Mathematics, 2013.
- 20 Thomas Kesselheim and Andreas Tönnis. Submodular Secretary Problems: Cardinality, Matching, and Linear Constraints. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2017)*, Leibniz International Proceedings in Informatics (LIPIcs), pages 16:1–16:22, 2017.
- 21 Robert Kleinberg. A Multiple-choice Secretary Algorithm with Applications to Online Auctions. In *Proceedings of the Sixteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '05, pages 630–631, Philadelphia, PA, USA, 2005. Society for Industrial and Applied Mathematics.
- 22 Nitish Korula, Vahab Mirrokni, and Morteza Zadimoghaddam. Online Submodular Welfare Maximization: Greedy Beats  $1/2$  in Random Order. In *Proceedings of the Forty-seventh Annual ACM Symposium on Theory of Computing*, STOC '15, pages 889–898. ACM, 2015.
- 23 D. V. Lindley. Dynamic Programming and Decision Theory. *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, 10(1):39–51, 1961.
- 24 Andrew McGregor and Hoa T Vu. Better Streaming Algorithms for the Maximum Coverage Problem. In *20th International Conference on Database Theory*, 2017.
- 25 Baharan Mirzasoleiman, Ashwinkumar Badanidiyuru, Amin Karbasi, Jan Vondrák, and Andreas Krause. Lazier Than Lazy Greedy. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, AAAI'15, pages 1812–1818. AAAI Press, 2015.
- 26 Martin Moser, Dusan P Jokanovic, and Norio Shiratori. An algorithm for the multidimensional multiple-choice knapsack problem. *IEICE transactions on fundamentals of electronics, communications and computer sciences*, 80(3):582–589, 1997.
- 27 George L Nemhauser and Laurence A Wolsey. Best algorithms for approximating the maximum of a submodular set function. *Mathematics of operations research*, 3(3):177–188, 1978.
- 28 Ashkan Norouzi-Fard, Jakub Tarnawski, Slobodan Mitrovic, Amir Zandieh, Aidasadat Mousavifar, and Ola Svensson. Beyond  $1/2$ -Approximation for Submodular Maximization on Massive Data Streams. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80, pages 3829–3838. PMLR, 10–15 jul 2018.
- 29 Cynthia A. Phillips, Cliff Stein, Eric Torng, and Joel Wein. Optimal time-critical scheduling via resource augmentation. *Algorithmica*, 32:163–200, 2001.
- 30 Robert J Vanderbei. The optimal choice of a subset of a population. *Mathematics of Operations Research*, 5(4):481–486, 1980.
- 31 Jan Vondrak. Optimal Approximation for the Submodular Welfare Problem in the Value Oracle Model. In *Proceedings of the Fortieth Annual ACM Symposium on Theory of Computing*, STOC '08, pages 67–74. ACM, 2008.
- 32 John G. Wilson. Optimal choice and assignment of the best  $m$  of  $n$  randomly arriving items. *Stochastic Processes and their Applications*, 39(2):325–343, 1991.
- 33 John G Wilson. Optimal choice and assignment of the best  $m$  of  $n$  randomly arriving items. *Stochastic processes and their applications*, 39(2):325–343, 1991.




# Hamiltonian Sparsification and Gap-Simulation

Dorit Aharonov<sup>1</sup>

School of Computer Science and Engineering, The Hebrew University, Jerusalem 91904, Israel  
dorit.aharonov@gmail.com

Leo Zhou<sup>2</sup>

Department of Physics, Harvard University, Cambridge, MA 02138, USA  
leozhou@g.harvard.edu

 <https://orcid.org/0000-0001-7598-8621>

---

## Abstract

Analog quantum simulation – simulation of one Hamiltonian by another – is one of the major goals in the noisy intermediate-scale quantum computation (NISQ) era, and has many applications in quantum complexity. We initiate the rigorous study of the physical resources required for such simulations, where we focus on the task of *Hamiltonian sparsification*. The goal is to find a simulating Hamiltonian  $\tilde{H}$  whose underlying interaction graph has bounded degree (this is called *degree-reduction*) or much fewer edges than that of the original Hamiltonian  $H$  (this is called *dilution*). We set this study in a relaxed framework for analog simulations that we call *gap-simulation*, where  $\tilde{H}$  is only required to simulate the groundstate(s) and spectral gap of  $H$  instead of its full spectrum, and we believe it is of independent interest.

Our main result is a proof that in stark contrast to the classical setting, general degree-reduction is *impossible* in the quantum world, even under our relaxed notion of gap-simulation. The impossibility proof relies on devising counterexample Hamiltonians and applying a strengthened variant of Hastings-Koma decay of correlations theorem. We also show a complementary result where degree-reduction is possible when the strength of interactions is allowed to grow polynomially. Furthermore, we prove the impossibility of the related sparsification task of generic Hamiltonian dilution, under a computational hardness assumption. We also clarify the (currently weak) implications of our results to the question of quantum PCP. Our work provides basic answers to many of the “first questions” one would ask about Hamiltonian sparsification and gap-simulation; we hope this serves as a good starting point for future research of these topics.

**2012 ACM Subject Classification** Theory of computation → Quantum computation theory

**Keywords and phrases** quantum simulation, quantum Hamiltonian complexity, sparsification, quantum PCP

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2019.2

**Related Version** Extended version including all proofs is hosted on <https://arxiv.org/abs/1804.11084>.

**Acknowledgements** We are grateful to Oded Kenneth for suggesting the construction of  $H_B$ , and for fruitful discussions; to Itai Arad for insightful remarks about the connection to quantum PCP; to Eli Ben-Sasson for discussions about PCP; to Ashley Montanaro and Toby Cubitt for clarifications about Hamiltonian simulation.

---

<sup>1</sup> Supported by ERC grant 280157, ISF grant No. 039-9494.

<sup>2</sup> Supported by ERC grant 280157, MIT MISTI.



## 1 Introduction

A major theme in quantum computation is the idea of *analog quantum simulation*. This is the task of simulating one Hamiltonian  $H$  by another Hamiltonian  $\tilde{H}$ , which might be more readily or easily implemented. In fact, this goal was identified as a main motivation for realizing quantum computers as early as 1981 by Feynman[37], with the idea that such analog quantum simulations can shed light on properties of physical quantum systems that are hard to simulate efficiently on classical computers. Cirac and Zoller [26] further developed this idea, and explained that such simulators are likely to be achievable well before fully fault-tolerant quantum computation [4, 48, 46] becomes practical, which might take a long time. While fault-tolerant quantum computers when realized can be used to apply *digital* quantum simulations [52] (where a quantum *circuit* simulates the time-evolution  $e^{-iHt}$  under a local Hamiltonian  $H$ ), *analog* quantum simulations are more accessible for near-term experiments because they do not require full-fledged quantum computer. Many groups are designing implementations in a variety of experimental platforms[59, 19, 18, 12, 42, 38], and we have recently witnessed some experiments in intermediate-sized quantum systems in regimes where classical simulations are difficult [17, 64, 44]. It has been argued that analog quantum simulation constitutes one of the more interesting challenges in the *noisy intermediate-scale quantum computing* (NISQ) era [55].

Beyond their natural physical applications, analog simulations of Hamiltonians are also very important for quantum complexity theory. For example, in the theory of quantum NP, one is often interested in *reducing* problems defined by one class of Hamiltonians to another (e.g. [47, 8, 22, 27, 45]). These reductions are often derived using *perturbative gadgets* (e.g. [45, 54, 21, 43, 24, 25]). Moreover, analog Hamiltonian simulators might also be useful for the design of Hamiltonian-based quantum algorithms, such as adiabatic algorithm [35] and QAOA [36]. In those settings, it is often desirable to tailor the Hamiltonians being used, while maintaining the properties essential for the algorithm.

In this paper, we initiate the rigorous study of the minimal resources required to simulate a given target Hamiltonian, and ask: When can we simulate a Hamiltonian  $H$  by another  $\tilde{H}$  that is *simpler*, easier, or more economic to implement? Of course, this vaguely stated question can take several forms if made more rigorous; here we focus on a natural goal which we loosely call *Hamiltonian sparsification*, which aims to simplify the *interaction graph* of the Hamiltonian. For a 2-local  $n$ -qubit Hamiltonian  $H$ , the interaction graph has  $n$  vertices, with edges connecting any pairs of qubits that participate in a local term in  $H$ . For a  $k$ -local Hamiltonian, we consider an interaction *hypergraph*, where each term acting on  $k$  qubits is represented by a hyperedge. A generic  $k$ -local Hamiltonian has  $\Theta(n^k)$  edges, and  $\Theta(n^{k-1})$  degree per vertex. Roughly speaking, Hamiltonian sparsification aims to simulate a Hamiltonian using another whose interaction graph is more “economic”, e.g., it has less edges (we refer to this as *dilution*) or its degree is bounded (we refer to this as *degree-reduction*).

Hamiltonian sparsification has several important motivations. First, it can help physicists tackle the immense hurdles they face when trying to realize Hamiltonians in the lab. In addition, in many settings in quantum complexity, such as in the study of quantum PCP [3] and recent approaches to the area law question [7], simulating a Hamiltonian by one with constant degree or fewer edges is a potentially important primitive. Indeed, sparsification is used ubiquitously in classical computer science, in a variety of applications; we mention two important ones. The first, graph sparsification (and more generally, matrix sparsification) is a central tool in matrix algorithms [61, 62, 60, 15]. Famously, Ref. [14] proved that any graph can be replaced by another which is sparse (namely, has small degree on average),

such that their Laplacian matrices are spectrally similar. Another common application of sparsification in classical computer science is *degree-reduction* (DR), used in the study of local Constraint Satisfaction Problems (CSPs) and PCPs [32]. We believe that this natural and basic notion deserves to be studied in the quantum context as well, and might have interesting applications beyond those we can foresee today.

### 1.1 Gap-Simulations: Simulating only the low-lying part of the spectrum

Before embarking on the study of Hamiltonian sparsification, we first need an appropriate definition of analog simulation. The study of analog Hamiltonian simulation was set on rigorous footing in a recent work by Cubitt, Montanaro, and Piddock [27]; their definition refines that of Bravyi and Hastings [22], and it roughly goes as follows: A given Hamiltonian  $H$  is simulated by “encoding” its full spectrum into the low-lying part of the spectrum of  $\tilde{H}$  acting on a larger Hilbert space. When  $\tilde{H}$  is implemented, then the low-lying part of its spectrum can be used to derive properties and information about the original Hamiltonian  $H$ . For obvious reasons, we will refer to this definition as *full-spectrum simulation*. In Ref. [27], the notion of *universal* Hamiltonians was defined and studied: these are families of Hamiltonians which are capable of performing full-spectrum simulations of *any* given Hamiltonian, albeit generally with exponential overhead in energy.

While this strong notion of *full-spectrum simulation* is necessary for simulating all dynamical properties of a system, it is common in physics that one is only interested in the properties of the low-energy states and, particularly, the groundstates. In addition, the spectral gap separating the groundstates from the rest of the spectrum is an intimately related quantity that is usually physically important. For example, the groundstates encode exotic quantum behaviors such as topological order, and the spectral gap protects them [63, 50]. Also, they are used together to define quantum phases of matter and characterize phase transitions [56, 28]. Moreover, both are the main objects of interest in quantum computational complexity: In quantum adiabatic algorithms [35], the goal is to prepare a groundstate of a problem Hamiltonian, and the spectral gap governs the efficiency of the process. In quantum NP theory [8], only the groundstate(s) of the Hamiltonian matters as it is the witness for the problem. The spectral gap also determines the temperature of a thermal equilibrium (Gibbs) state that can be used to approximate the groundstate. Hence, we believe that a natural and minimal notion of analog Hamiltonian simulation, which is still meaningful for many physical contexts, should require that both the space of groundstates and the spectral gap above it be preserved.

Therefore, we suggest to consider sparsification, or more generally Hamiltonian simulation, using this minimal notion, which we formally define as *gap-simulation*. To the best of our knowledge, despite its naturalness, this relaxed notion of Hamiltonian simulation was not formally defined and rigorously studied previously in the quantum complexity literature.

A Hamiltonian  $\tilde{H}$  is said to *gap-simulate*  $H$  if it mimics the groundstate(s) and the spectral gap of  $H$ ; no constraints are imposed on the excited part of the spectrum. To provide a sensible definition requires some care, since in the quantum world we can allow inaccuracies and entanglement to an ancilla. We provide two versions of the definition: In the weaker one (Definition 3), the groundspace is mimicked *faithfully*, i.e. the *support* of any groundstate of  $\tilde{H}$ , when reduced to the original Hilbert space, is close to the groundspace of  $H$ . However, this definition does not require quantum *coherence* within the groundspace be maintained. Such coherence is guaranteed by our stronger definition (Definition 2), in which all superpositions within the groundspace are simulated. The extent to which the gap-simulation is incoherent (or unfaithful) is quantified via a small constant  $\epsilon$  (or  $\delta$ ). It

seems that the coherent notion is the “correct” one for most quantum applications, though the weaker one might also be useful in certain contexts (see Sec. 5). We mention that here, like in Ref. [22, 27], we allow encoding of the qubits. Typically, we consider “localized” encodings, though this is not explicitly required in the definition.

To set the stage, some basic results about the framework are provided: We show in Lemma 4 that for Hamiltonians with unique groundstates, our two definitions of gap-simulations coincide. Moreover, both coherent and incoherent gap-simulation definitions are shown to be stable under compositions (Lemma 6).

How does the gap-simulation framework compare with the stricter definitions of full-spectrum simulations developed in Ref. [22, 27]? In Sec. 2.1.1, this connection is discussed formally; roughly, our definition is indeed a relaxed version of full-spectrum simulations whose spectral error is smaller than the spectral gap, up to varying restrictions about encoding (Lemma 8). We choose to work here with the more relaxed definition of gap-simulation, since impossibility results for a weaker definition are of course stronger. More generally, it seems that this framework is an important and relevant one to consider in physics and quantum complexity contexts. Being less demanding, gap-simulation is likely achievable in certain cases where full-spectrum simulation is difficult or even impossible.

## 1.2 Main Results

Equipped with this framework of Hamiltonian sparsification via gap-simulations, we ask: When are sparsifications possible in the quantum world? It is conceivable that, like in the analogous classical settings mentioned above [32, 15], they ought to be always possible. The main result of in this paper (Theorem 11) shows that in stark contrast to the classical setting, both coherent and incoherent degree-reductions are not generally possible in the quantum world, even if one uses the relaxed notion of gap-simulation. This impossibility phenomenon is due to the existence of many-body entanglement in some quantum groundstates; we show, using a strengthened version of Hastings-Koma decay of correlation theorem [41], that there exist local Hamiltonians whose groundstates cannot be coherently mapped into the groundspace of a gapped Hamiltonian with constant degree and bounded interaction strength. Though one might suspect this is a consequence of degeneracy in the groundspace, we show that it holds even in the case of a unique groundstate. We believe this is a surprising and curious phenomenon, which demonstrates the richness in the subject, and highlights the difference in the resources required for classical versus quantum Hamiltonian simulation.

This impossibility result on degree-reduction is essentially tight, as we provide a complementary result (Theorem 13) based on a somewhat sophisticated application of the circuit-to-Hamiltonian construction, stating that degree-reduction becomes possible for any local Hamiltonian with non-negligible spectral gap, when polynomially large overhead in interaction strength is allowed.

We also study a related important sparsification task: dilution. While our main result of Theorem 11 is an information-theoretic result that rules out *existence* of degree-reducers regardless of computational power, we are unable to provide such a strong result in the case of dilution. Information-theoretically, we can only rule out dilution with perfect (or inverse-polynomially close to perfect) coherence (Theorem 15). Nevertheless, we are able to prove impossibility of any efficient classical algorithm to find diluters with constant unfaithfulness, for generic (even classical) Hamiltonians (Theorem 16). The proof of this theorem (relying on Ref. [29]) works under the assumption that  $\text{coNP} \not\subseteq \text{NP/poly}$  (alternatively, the polynomial hierarchy does not collapse to its third level). Although generic constructive dilution is ruled out by our Theorem 16, the question of existence of diluters for general Hamiltonian, with bounded or large interaction strengths, remains open.



The paper provides quite a few further results complementing the above-mentioned main contributions. These build on ideas in classical PCP reductions and perturbative gadgets. In addition, the ideas studied here are strongly reminiscent of questions arising in the context of the major open problem of quantum PCP [3]. We clarify this connection and provide some preliminary results along those lines.

We believe that the study of the resources required for Hamiltonian simulations in various contexts, as well as the framework of gap-simulation, are of potential deep interest for physics as well as quantum complexity. The questions raised touch upon a variety of important challenges, from quantum simulations, to algorithm design, to quantum PCP and NP reductions, to physical implementations on near-term quantum processors, and more. Their study might also shed light on questions in many-body physics, by developing tools to construct “equivalent” Hamiltonians, from the point of view of the study of groundstate physics. The discussion in Sec. 5 includes a more detailed list of some of the open questions and implications.

### 1.3 Overview

In Sec. 2, we set the stage by providing definitions of gap-simulation and sparsification, and proving basic facts about this new framework. In Sec. 3, we state our results formally. Subsequently, Sec. 4 provides elaborated and intuitive proof sketches, and Sec. 5 provides further discussion. All technical proofs are deferred to the extended version [1].

## 2 Definition of the Framework: Setting the Stage

### 2.1 Gap-Simulations of Hamiltonians

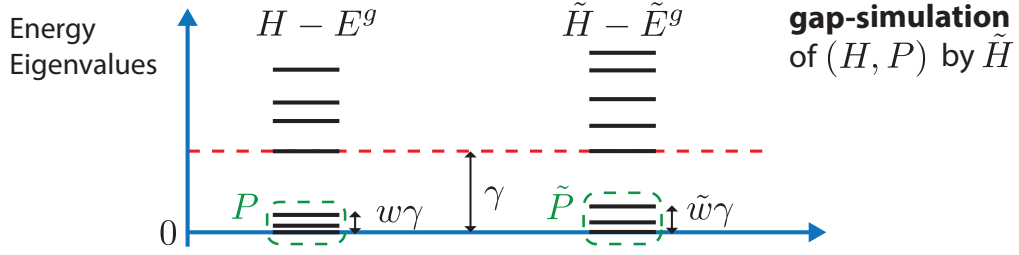
We restrict our attention to  $k$ -local Hamiltonians  $H = \sum_{i=1}^M H_i$  acting on  $n$  qudits (with internal states  $\{|0\rangle, \dots, |d-1\rangle\}$ ), where each term  $H_i$  acts nontrivially on a (distinct) subset of at most  $k$  qudits. We denote  $\lambda_j(X)$  as the  $j$ -th lowest eigenvalue of  $X$ , and  $\|X\|$  as the spectral norm of  $X$ . In addition, for any Hermitian projector  $P$ , we denote  $P^\perp \equiv \mathbb{1} - P$ , and  $|\psi\rangle \in P \iff P|\psi\rangle = |\psi\rangle$ .

► **Definition 1** (groundspace, energy spread and gap). Consider a family of  $n$ -qudit Hamiltonians  $\{H_{(n)}\}_{n=1}^\infty$ . Let  $E_n^g = \lambda_1(H_{(n)})$ , and suppose  $P_{(n)}$  is a Hermitian projector onto the subspace of eigenstates of  $H_{(n)}$  with energy  $\leq E_n^g + w_n\gamma_n$ , for some  $\gamma_n > 0$ ,  $0 \leq w_n < 1$ , such that

$$\begin{aligned} [H_{(n)}, P_{(n)}] &= 0, & \|P_{(n)}(H_{(n)} - E_n^g)P_{(n)}\| &\leq w_n\gamma_n, \\ \text{and } \lambda_j(P_{(n)}^\perp(H_{(n)} - E_n^g)P_{(n)}^\perp + \gamma_n P_{(n)}) &\geq \gamma_n & \forall j. \end{aligned} \tag{1}$$

We call the subspace onto which  $P_{(n)}$  projects a *quasi-groundspace*,  $w_n$  its *energy spread*, and  $\gamma_n$  its *quasi-spectral gap*. When we choose  $w_n = 0$  and  $\gamma_n = \min_j \{\lambda_j(H_{(n)}) - E_n^g : \lambda_j(H_{(n)}) \neq E_n^g\}$ , we call the quasi-groundspace that  $P_{(n)}$  projects onto simply *the groundspace* of  $H_{(n)}$ , and  $\gamma_n$  *the spectral gap* of  $H_{(n)}$ . Let  $w_\infty = \sup_n w_n$  and  $\gamma_\infty = \inf_n \gamma_n$ . If  $\gamma_\infty > 0$  and  $w_\infty < 1$ , we say  $\{H_{(n)}\}_{n=1}^\infty$  is *spectrally gapped*.

Below, we omit the subscript  $n$  in  $H_{(n)}$ , referring to a single  $H$ , with the usual implicit understanding that we consider families of Hamiltonians, where  $n \rightarrow \infty$ . All explicit Hamiltonians we gap-simulate here have  $w_n = 0$ , but Definition 1 is more general and allows  $w_n > 0$ , so that it can capture situations with slightly perturbed groundstates (or when simulating a larger low-energy part of the spectrum). We now define Hamiltonian gap-simulation, visualized in Fig. 1:



■ **Figure 1** Visualizing gap-simulation of Hamiltonian  $H$  with quasi-groundspace projector  $P$  by  $\tilde{H}$ . If  $\|\tilde{P} - V(P \otimes P_{\text{anc}})V^\dagger\| \leq \epsilon$ , for some isometry  $V$ , then this is a coherent gap-simulation with  $\epsilon$ -incoherence. If  $\|\tilde{P} - V(P \otimes \mathbb{1}_{\text{anc}})V^\dagger\tilde{P}\| \leq \delta$ , then this is an incoherent but faithful gap-simulation with  $\delta$ -unfaithfulness.

► **Definition 2** (gap-simulation of Hamiltonian). Let  $H$  and  $\tilde{H}$  be two Hamiltonians, defined on Hilbert spaces  $\mathcal{H}$  and  $\tilde{\mathcal{H}}$  respectively, where  $\tilde{\mathcal{H}}$ . Let  $V : \mathcal{H} \otimes \mathcal{H}_{\text{anc}} \rightarrow \tilde{\mathcal{H}}$  be an isometry ( $V^\dagger V = \mathbb{1}$ ), where  $\mathcal{H}_{\text{anc}}$  is some ancilla Hilbert space. Denote  $\tilde{E}^g \equiv \lambda_1(\tilde{H})$ . Per Definition 1, let  $P$  be a quasi-groundspace projector of  $H$ ,  $\gamma$  its quasi-spectral gap. We say that  $\tilde{H}$  *gap-simulates*  $(H, P)$  with *encoding*  $V$ , *incoherence*  $\epsilon \geq 0$  and *energy spread*  $0 \leq \tilde{w} < 1$  if the following conditions are both satisfied:

1. There exists a Hermitian projector  $\tilde{P}$  onto a subspace of eigenstates of  $\tilde{H}$  such that

$$[\tilde{H}, \tilde{P}] = 0, \quad \|\tilde{P}(\tilde{H} - \tilde{E}^g)\tilde{P}\| \leq \tilde{w}\gamma, \quad \text{and} \quad \lambda_j(\tilde{P}^\perp(\tilde{H} - \tilde{E}^g)\tilde{P}^\perp + \gamma\tilde{P}) \geq \gamma \quad \forall j. \quad (2)$$

I.e.,  $\tilde{P}$  projects onto a quasi-groundspace of  $\tilde{H}$  with quasi-spectral gap not smaller than that of  $P$  in  $H$ , and energy spread  $\tilde{w}$ .

2. There exists a Hermitian projector  $P_{\text{anc}}$  acting on  $\mathcal{H}_{\text{anc}}$ , so that

$$[\text{bounded incoherence}] \quad \|\tilde{P} - V(P \otimes P_{\text{anc}})V^\dagger\| \leq \epsilon. \quad (3)$$

When  $P$  projects onto the groundspace of  $H$ , rather than a quasi-groundspace, we usually do not mention  $P$  explicitly, and simply say that  $\tilde{H}$  *gap-simulates*  $H$ .

Requiring  $\epsilon$  from Eq. (3) be small ensures that *coherence* in the groundspace is maintained by the gap-simulation. This is illustrated by considering a Hamiltonian  $H$  with two orthogonal groundstates  $|g_1\rangle$  and  $|g_2\rangle$ . The condition of Eq. (3) essentially says that for any coherent superposition  $|g\rangle = c_1|g_1\rangle + c_2|g_2\rangle$ , and a state  $|a\rangle \in P_{\text{anc}}$  on the ancilla, there exists a groundstate of  $\tilde{H}$  that looks like  $|\tilde{g}\rangle = V(|g\rangle \otimes |a\rangle) + \mathcal{O}(\epsilon)$ . Moreover, any groundstate of  $\tilde{H}$  could be written in this form. This would preserve the expectation value of any observable in the groundspace, i.e.  $\langle g|\hat{\sigma}|g\rangle \approx \langle \tilde{g}|V\hat{\sigma}V^\dagger|\tilde{g}\rangle + \mathcal{O}(\epsilon)$ . In contrast, one can consider an alternative situation where the groundspace of a simulator  $\tilde{H}$  is spanned by states of the form  $|\tilde{g}'_i\rangle \approx V(|g_i\rangle \otimes |a_i\rangle)$ , where  $\langle a_i|a_j\rangle \ll 1$ . This situation remains interesting, as finding a groundstate  $|\tilde{g}'_i\rangle$  of  $\tilde{H}$  reveals information about a groundstate of  $H$  by decoding:  $|g_i\rangle\langle g_i| \approx \text{Tr}_{\text{anc}}(V^\dagger|\tilde{g}'_i\rangle\langle\tilde{g}'_i|V)$ . However, the coherence among groundstates is destroyed, since  $|g\rangle = |g_1\rangle + |g_2\rangle$  is mapped to  $|\tilde{g}'\rangle \approx V(|g_1\rangle \otimes |a_1\rangle + |g_2\rangle \otimes |a_2\rangle)$ , and observables such as  $\hat{\sigma} = |g_1\rangle\langle g_2|$  are not preserved:  $\langle g|\hat{\sigma}|g\rangle \not\approx \langle \tilde{g}'|V\hat{\sigma}V^\dagger|\tilde{g}'\rangle$ .

Although coherence seems important to maintain in most quantum settings, we also define *incoherent* gap-simulation, which may be relevant in some situations (see discussion in Sec. 5).

► **Definition 3** (incoherent gap-simulation). Consider two Hamiltonians  $H$  and  $\tilde{H}$ ,  $P$  a quasi-groundspace projector of  $H$ , and  $V$  some isometry in the same setting as in Definition 2. We say that  $\tilde{H}$  *incoherently gap-simulates*  $(H, P)$  with encoding  $V$ , unfaithfulness  $\delta \geq 0$  and energy spread  $0 \leq \tilde{w} < 1$  if it satisfies the first condition of Definition 2 and, instead of the second condition of Eq. (3),

$$[\text{bounded unfaithfulness}] \quad \|\tilde{P} - V(P \otimes \mathbf{1}_{\text{anc}})V^\dagger \tilde{P}\| \leq \delta. \quad (4)$$

Again, when  $P$  projects onto the groundspace of  $H$ , we simply say  $\tilde{H}$  *incoherently gap-simulates*  $H$ .

Small unfaithfulness essentially means that the *support* of the vectors in the groundspace of  $\tilde{H}$  is roughly contained in a subspace spanned by encoding the groundspace of  $H$  with some ancilla.

It is easy to see that small incoherence implies small unfaithfulness, namely  $\delta \leq 2\epsilon$  [1]. However, small unfaithfulness is a strictly weaker condition than small incoherence; we will see an example in Prop. 17. Importantly, when  $H$  has a *unique* groundstate, the two notions are equivalent up to a constant (the proof of this fact is surprisingly not trivial; see Sec. 4.2):

► **Lemma 4** (incoherent gap-simulation is coherent when groundstate is unique). *Suppose  $H$  has a unique groundstate, with groundspace projector  $P = |g\rangle\langle g|$ . If  $\tilde{H}$  incoherently gap-simulates  $H$  with unfaithfulness  $\delta < 1$ , then it also gap-simulates  $H$  with incoherence  $\epsilon \leq \sqrt{2\delta}/\sqrt{1-\delta^2}$ .*

While we do not explicitly restrict the form of encoding  $V$  in the above definitions, we need to specify them for the impossibility proofs, where we will consider localized encoding:

► **Definition 5** (localized encoding). Consider a (possibly incoherent) gap-simulation of  $H$  by  $\tilde{H}$  encoded by an isometry  $V : \mathcal{H} \otimes \mathcal{H}_{\text{anc}} \rightarrow \tilde{\mathcal{H}}$ . Let  $\mathcal{H} \otimes \mathcal{H}_{\text{anc}} = [\bigotimes_{i=1}^n (\mathcal{H}_i \otimes \mathcal{A}_i)] \otimes \mathcal{A}_E$ , where  $\mathcal{H}_i$  is the  $i$ -th qudit in  $H$ ,  $\mathcal{A}_i$  is its associated ancilla subsystem, and  $\mathcal{A}_E$  contains the remaining ancillas; also let  $\tilde{\mathcal{H}} = \bigotimes_{i=1}^m \tilde{\mathcal{H}}_i$ ,  $m \geq n$ . We say  $V$  is a *localized encoding* if either of the following is true:

1.  $V = [\bigotimes_{i=1}^n V_i] \otimes V_E$ , where  $V_i : \mathcal{H}_i \otimes \mathcal{A}_i \rightarrow \tilde{\mathcal{H}}_i$  is an isometry, and  $\tilde{\mathcal{H}}_i$  consists of  $O(1)$  qudits in  $\tilde{H}$  for  $i = 1, \dots, n$ . Also,  $V_E : \mathcal{A}_E \rightarrow \bigotimes_{i=n+1}^m \tilde{\mathcal{H}}_i$  is an isometry.
2.  $V$  is a constant-depth quantum circuit:  $V = \prod_{a=1}^D U_a$ , where  $D = O(1)$ ,  $U_a = \bigotimes_{\mu} U_{a,\mu}$ , and  $U_{a,\mu}$  is a unitary operator acting on  $O(1)$  number of qudits.

We say  $V$  is an  $\eta$ -*localized encoding* if there is a localized encoding  $V_L$  such that  $\|V - V_L\| \leq \eta$ .

In addition to constant-depth quantum circuits, any quantum error-correcting code where each logical qudit is encoded as  $O(1)$  qudits is also a localized encoding. Note it is easy to see that if a gap-simulation has  $\eta$ -localized encoding  $V$  and incoherence  $\epsilon$  (or unfaithfulness  $\delta$ ), it is also a gap-simulation with localized encoding  $V_L$  and incoherence  $\epsilon' \leq \epsilon + 2\eta$  (or unfaithfulness  $\delta' \leq \delta + 2\eta$ ). Hence, we usually restrict our attention to fully localized encoding in the remainder of the paper.

It is also fairly straightforward to show that compositions of gap-simulation behave intuitively:

► **Lemma 6** (Composition). *Suppose  $H_1$  (incoherently) gap-simulates  $(H_0, P_0)$  with encoding  $V_1$ , incoherence  $\epsilon_1$  (or unfaithfulness  $\delta_1$ ), energy spread  $\tilde{w}_1$ , and a corresponding quasi-groundspace projector  $P_1$ . Also suppose  $H_2$  (incoherently) gap-simulates  $(H_1, P_1)$  with encoding  $V_2$ , incoherence  $\epsilon_2$  (or unfaithfulness  $\delta_2$ ), and energy spread  $\tilde{w}_2$ . Then  $H_2$  (incoherently) gap-simulates  $(H_0, P_0)$  with encoding  $V_2(V_1 \otimes \mathbf{1}_{\text{anc},1})$ , incoherence  $\leq \epsilon_2 + \epsilon_1$  (or unfaithfulness  $\leq 2\delta_2 + \delta_1$ ), and energy spread  $\tilde{w}_2$ .*

### 2.1.1 Comparison of gap-simulation to full-spectrum simulation

To formally compare gap-simulations to full-spectrum simulations of Hamiltonians, we use the following well-motivated definition of full-spectrum simulation developed by Ref. [27]:

► **Definition 7** (Full-spectrum simulation, adapted from Definition 1 of [27]). A Hamiltonian  $\tilde{H}$  *full-spectrum-simulates* a Hamiltonian  $H$  to precision  $(\eta, \xi)$  below an energy cut-off  $\Delta$  if there exists an encoding  $\mathcal{E}(H) = V(H \otimes P + \bar{H} \otimes Q)V^\dagger$ , where  $V$  is an isometry,  $P$  and  $Q$  are mutually orthogonal projectors, such that

1. There exists an encoding  $\tilde{\mathcal{E}}(H) = \tilde{V}(H \otimes P + \bar{H} \otimes Q)\tilde{V}^\dagger$  such that  $\|\tilde{V} - V\| \leq \eta$  and  $\tilde{\mathcal{E}}(\mathbb{1}) = P_{\leq \Delta(\tilde{H})}$ ;
2.  $\|\tilde{H}_{\leq \Delta} - \tilde{\mathcal{E}}(H)\| \leq \xi$ .

Here,  $P_{\leq \Delta(\tilde{H})}$  projects onto eigenstates of  $\tilde{H}$  with eigenvalues  $\leq \Delta$ , and  $\tilde{H}_{\leq \Delta} = \tilde{H}P_{\leq \Delta(\tilde{H})}$ .

The appearance of  $\bar{H}$ , the complex-conjugate of  $H$ , is necessary to allow for encoding of complex Hamiltonians into real ones. Note that for any real-valued Hamiltonian  $H$ , we can simply write  $\mathcal{E}(H) = V(H \otimes P_{\text{anc}})V^\dagger$ , where  $P_{\text{anc}} = P + Q$  is a projector since  $P$  and  $Q$  are orthogonal.

In the main definition used by Ref. [27], they have also motivated a natural restriction that the encoding should be *local*. Specifically, their restriction to “local encoding” requires that  $V = \bigotimes_i V_i$  for some isometries  $V_i$  each acting on at most one qudit from  $H$ ; moreover,  $P$  and  $Q$  are locally orthogonal projectors, in the sense that there are projectors  $P_i \perp Q_i$  acting on the same qudits as  $V_i$  where  $P_i P = P$  and  $Q_i Q = Q$ . We note that our localized encodings per Definition 5 is somewhat different than this notion of “local encoding”. For example, constant-depth circuit qualifies as a localized encoding but not a “local encoding”, due to the possibility of overlaps between supports of encoded qubits (and hence  $V$  cannot be written in tensor-product form). On the other hand, Ref. [27] does not place any explicit restriction on the size of the support of each encoded qubit in their definition. Nevertheless, the specific encodings utilized for simulating general spin-Hamiltonians by universal Hamiltonians in [27] all have  $O(1)$ -sized supports, and thus also qualify as localized encodings.

It is not difficult to show that full-spectrum simulation by Definition 7 with sufficiently small precision ( $\xi \ll (1-w)\gamma$ ) implies a coherent gap-simulation by our Definition 2, if we restrict to encodings of the form  $\mathcal{E}(H) = V(H \otimes P_{\text{anc}})V^\dagger$ . This restriction of the encoding format simplifies the comparison, and has no loss of generality when considering real-valued Hamiltonians.

► **Lemma 8** (Full-spectrum simulation implies coherent gap-simulation). *Let  $H$  be a Hamiltonian that has a quasi-groundspace projector  $P$  with quasi-spectral gap  $\gamma$  and energy spread  $w \leq 1/2$ . Suppose  $\tilde{H}$  full-spectrum-simulates  $H$  to precision  $(\eta, \xi)$  according to Definition 7 with encoding  $\mathcal{E}(H) = V(H \otimes P_{\text{anc}})V^\dagger$ , such that  $\xi \leq (1-w)\gamma/8$ . Then  $\tilde{H}' = \frac{4}{3}\tilde{H}$  gap-simulates  $(H, P)$  with encoding  $V$ , incoherence  $\epsilon \leq 32\xi/\gamma + 2\eta$ , and energy spread  $\tilde{w} \leq (w + 2\xi/\gamma)/(1 - 2\xi/\gamma)$ , per Definition 2.*

## 2.2 Hamiltonian Sparsification: Degree-Reduction and Dilution

We define here the set of parameters of interest when considering minimizing resources in gap-simulations:

1.  $k$  – locality of individual Hamiltonian term; typically  $O(1)$  in physical systems, but we parametrize it to allow minimization, as well as to allow  $O(\log^a n)$ -local Hamiltonians, for some constant  $a$ .

2.  $r$  – maximum degree of Hamiltonian, the main objective in degree-reduction.
3.  $M$  – number of terms in the Hamiltonian, the main objective in dilution.
4.  $J$  – the interaction strength of individual Hamiltonian terms. This is typically restricted to  $O(1)$  in physical systems, but allowing it to grow with  $n$  leads to more possibilities of gap-simulators. Equivalently, a gap-simulator with  $J$  growing with  $n$  can be converted to one that simulates the original Hamiltonian but has a vanishing gap if we restrict to bounded-strength Hamiltonian terms.
5.  $\epsilon$  and  $\delta$  – incoherence  $\epsilon$  and unfaithfulness  $\delta$  that capture how well the Hamiltonian gap-simulates the original Hamiltonian in terms of groundspace projectors.
6.  $\tilde{w}$  – energy spread in the gap-simulator Hamiltonian; allowing it to be different from the original Hamiltonian gives more freedom in gap-simulations.

We will use the notation of  $[r, M, J]$ -gap-simulator to indicate that the maximum degree is  $r$ , the number of local terms is  $M$ , and for each term  $\tilde{H}_i$  we have  $\|\tilde{H}_i\| \leq J$ . We define:

► **Definition 9** (Degree-reduction (DR) and dilution). Let  $\tilde{H}$  be a  $k$ -local  $[r, M, J]$ -gap-simulator of  $H$  with  $\epsilon$ -incoherence (or  $\delta$ -unfaithfulness) and energy spread  $\tilde{w}$ . Additionally suppose  $H = \sum_{i=1}^{M_0} H_i$  is a sum of  $M_0 = M_0(n)$  terms, each of which is  $O(1)$ -local. Then

- We call  $\tilde{H}$  an  $[r, M, J]$ -degree-reducer of  $H$  if  $r = O(1)$ .
- We call  $\tilde{H}$  an  $[r, M, J]$ -diluter of  $H$  if  $M = o(M_0(n))$ .

We also call any degree-reducer or diluter of  $H$  a *sparsifier* of  $H$ .

### 3 Results

Our impossibility results are based on two families of 2-local  $n$  qubits Hamiltonians, which can both be expressed in terms of the collective angular momentum operator  $\mathcal{J}_\alpha = \sum_{i=1}^n \sigma_\alpha^{(i)}/2$  for  $\alpha \in \{x, y, z\}$ , where  $\sigma_\alpha^i$  are the standard Pauli matrices.

► **Example A** (degenerate groundstates).

$$H_A = \left(\mathcal{J}_z + \frac{n}{2}\right) \left(\mathcal{J}_z + \frac{n}{2} - 1\right) = \frac{1}{4} \sum_{i < j}^n (1 - \sigma_z^{(i)}) \otimes (1 - \sigma_z^{(j)}) = \sum_{i < j}^n |1\rangle\langle 1|^{(i)} \otimes |1\rangle\langle 1|^{(j)}. \quad (5)$$

There are  $M_0(n) = \frac{1}{2}n(n-1)$  terms in  $H_A$ , and each qubit has degree  $n-1$ . The terms in  $H_A$  mutually commute, and its groundspace is spanned by the following  $n+1$  zero-energy orthonormal states that have  $\mathcal{J}_z = -n/2$  or  $\mathcal{J}_z = -n/2 + 1$ :

$$GS(H_A) = \text{span}\{|00 \cdots 00\rangle, |00 \cdots 01\rangle, |00 \cdots 10\rangle, \dots, |10 \cdots 00\rangle\}. \quad (6)$$

If we consider a qubit in  $|1\rangle$  to be an “excitation,” the groundstates are states with one or zero “excitations.” Observe that  $w_n = 0$  and  $\gamma_n = 1$ , independent of  $n$ ; the system is thus spectrally gapped.

► **Example B** (unique groundstate). In this example we require that  $n$  is even,  $n = 2s$ :

$$\begin{aligned} H_B &= \mathcal{J}_z^2 - \frac{1}{2}\vec{\mathcal{J}}^2 + b_n = \frac{1}{2}(\mathcal{J}_z^2 - \mathcal{J}_x^2 - \mathcal{J}_y^2) + b_n \\ &= \frac{1}{4} \sum_{i < j}^n (\sigma_z^{(i)} \sigma_z^{(j)} - \sigma_x^{(i)} \sigma_x^{(j)} - \sigma_y^{(i)} \sigma_y^{(j)}) - \frac{n}{8} + b_n \end{aligned} \quad (7)$$

where  $b_n \equiv \frac{1}{8}n(n+2)$  is a constant chosen so that  $\lambda_1(H_B) = 0$ . Similarly to  $H_A$ , this Hamiltonian has  $M_0(n) = \frac{1}{2}n(n-1)$  2-local terms, and each qubit has degree  $n-1$ . Since  $[\tilde{\mathcal{J}}^2, \mathcal{J}_z] = 0$ , the eigenstates of  $H_B$  can be written in eigenbasis of both  $\tilde{\mathcal{J}}^2$  and  $\mathcal{J}_z$ . As  $\mathcal{J}_\alpha$  are spin- $s$  angular momentum operators,  $\mathcal{J}_z^2$  has eigenvalues  $\{0, 1, 2^2, \dots, s^2\}$  and  $\tilde{\mathcal{J}}^2$  has eigenvalues  $\{s(s+1), (s-1)s, \dots, 6, 2, 0\}$ . The groundstate of  $H_B$  is thus a state that has minimal  $\mathcal{J}_z^2 = 0$  and maximal total angular momentum  $\mathcal{J} = s = \frac{n}{2}$ . This is a unique state, which is known as a Dicke state [30]:

$$|g_B\rangle = |\mathcal{J} = \frac{n}{2}; \mathcal{J}_z = 0\rangle = \binom{n}{n/2}^{-1/2} \sum_{|\{i : x_i = 1\}| = n/2} |x_1 \cdots x_n\rangle. \quad (8)$$

where the state can be explicitly written as a symmetric superposition of all strings  $x$  with Hamming weight  $h(x) = |\{i : x_i = 1\}| = n/2$ . This groundstate  $|g_B\rangle$  has energy 0. Meanwhile, all other eigenstates must have energy at least 1. Thus, the system is spectrally gapped with energy spread  $w_n = 0$  and  $\gamma_n = 1$ .

It turns out that these deceptively simple examples form a challenge for Hamiltonian sparsification.

### 3.1 Limitations on Degree-Reduction

For didactic reasons, we start by ruling out generic *perfectly coherent* DR. This is done by showing that such DR is impossible for  $H_A$ .

► **Lemma 10** (Impossibility of generic 0-incoherence DR). *There does not exist any  $k$ -local Hamiltonian  $\tilde{H}_A$  that is an  $[o(n/k), M, J]$ -degree-reducer of the  $n$ -qubit Hamiltonian  $H_A$  with localized encoding, 0-incoherence, and energy spread  $\tilde{w} < 1/2$ , regardless of number of terms  $M$  or interaction strength  $J$ .*

A closer inspection of the proof implies a trade-off between  $\epsilon$  and  $J$ , from which it follows that if  $J = O(1)$  then generic DR is impossible even if we allow  $\epsilon$  which is inverse polynomially small (see exact statement in extended version [1]). We note that this result in fact rules out any improvement of the degree for  $H_A$ , to some sub-linear degree.

However, perfect (or even inverse-polynomially close to perfect) coherence is a rather strong requirement. Indeed, by improving our proof techniques, we manage to improve our results for  $H_A$  to show impossibility even for constant coherence. Moreover, by devising another Hamiltonian with a unique groundstate,  $H_B$ , and proving such an impossibility result also for this Hamiltonian, we arrive at the following theorem. Our main result is a strong impossibility result, ruling out generic DR with *constant unfaithfulness* (and consequently, also constant incoherence).

► **Theorem 11** (Main: Impossibility of constant coherence (faithfulness) DR for  $H_A$  ( $H_B$ )). *For sufficiently small constants  $\epsilon \geq 0$  ( $\delta \geq 0$ ) and  $\tilde{w} \geq 0$ , there exists system size  $n_0$  where for any  $n \geq n_0$ , there is no  $O(1)$ -local  $[O(1), M, O(1)]$ -degree-reducer of the  $n$ -qubit Hamiltonian  $H_A$  ( $H_B$ ) with localized encoding,  $\epsilon$ -incoherence ( $\delta$ -unfaithfulness), and energy spread  $\tilde{w}$ , for any number of Hamiltonian terms  $M$ .*

We deduce that generic quantum DR, with even constant unfaithfulness, is impossible. This stands in striking contrast to the classical setting. It is well known that classical DR is possible for all CSPs in the context of PCP reductions[32]. This construction easily translates to a 0-unfaithfulness degree-reducer for any *classical* local Hamiltonian:

► **Proposition 12** (Incoherent DR of classical Hamiltonians). *Consider an  $n$ -qudit  $k$ -local classical Hamiltonian  $H = \sum_{S \subset \{1, \dots, n\}} C_S$ , where each  $C_S : \{z_i : i \in S\} \rightarrow [0, 1]$  is a function of  $d$ -ary strings of length  $|S| \leq k$  representing states of qudits in  $S$ . Let the number of terms in  $H$  be  $M_0 = |\{S\}| = O(n^k)$ . Then there is a  $k$ -local  $[3, O(kM_0), O(1)]$ -degree-reducer of  $H$  with 0-unfaithfulness, energy spread  $\tilde{w} = 0$ , and trivial encoding  $V = \mathbb{1}$ .*

This demonstrates a large difference between the quantum and classical settings in the context of Hamiltonian sparsification. Characterizing which quantum Hamiltonians can be degree-reduced (with bounded interaction strength), either coherently or just faithfully, remains open.

The impossibility of DR by Theorem 11, which heavily relies on the interaction strength  $J$  being a constant, is essentially tight. We prove this in a complementary result showing that degree-reduction is possible when  $J$  is allowed to grow polynomially for any local Hamiltonian whose spectral gap closes slower than some polynomial (which is the case of interest for gap-simulation):

► **Theorem 13** (Coherent DR with polynomial interaction strength). *Suppose  $H$  is an  $O(1)$ -local Hamiltonian with a quasi-groundspace projector  $P$ , which has quasi-spectral gap  $\gamma = \Omega(1/\text{poly}(n))$  and energy spread  $w$ . Also assume  $\|H\| = O(\text{poly}(n))$ . Then for any  $\epsilon > 0$ , one can construct an  $O(1)$ -local  $[O(1), O(\text{poly}(n)/\epsilon^2), O(\text{poly}(n, \epsilon^{-1}))]$ -degree-reducer of  $H$  with incoherence  $\epsilon$ , energy spread  $w + O(1/\text{poly}(n))$ , and trivial encoding.*

The proof is constructive: we map any given Hamiltonian to the quantum phase-estimation circuit, make the circuit sparse with new ancilla qudits and swap gates, and transform it back to a Hamiltonian using Kitaev's circuit-to-Hamiltonian construction [47]. Some innovations are required to ensure coherence within the groundspace isn't destroyed. For the most general local Hamiltonian whose spectral gap may close exponentially (or possibly even faster, see e.g. [2]), we can show that coherent DR is possible with interaction strength that scales exponentially with inverse gap and incoherence:

► **Theorem 14** (Coherent DR with exponential interaction strength). *Let  $H$  be an  $n$ -qubit  $O(1)$ -local Hamiltonian with  $M_0$  terms, each with bounded norm. Suppose  $H$  has quasi-spectral gap  $\gamma$  and energy spread  $w$ . For any  $\epsilon > 0$ , one can construct a 2-local  $[O(1), O(M_0), O((\gamma\epsilon)^{-\text{poly}(n)})]$ -degree-reducer of  $H$  with incoherence  $\epsilon$ , energy spread  $w + \mathcal{O}(\epsilon)$ , and trivial encoding.*

The proof uses a construction from perturbative gadgets, and is similar to other results in the Hamiltonian simulation literature [54, 27]. Due to significantly more resource required compared to Theorem 13, this construction is only useful in situations where we want to preserve some extremely small spectral gap.

### 3.2 Limitations on Dilution

For perfect or near-perfect dilution, we can prove a similar impossibility result to Lemma 10:

► **Theorem 15** (Impossibility of generic 0-incoherence dilution). *There does not exist any  $k$ -local Hamiltonian  $\tilde{H}_A$  that is an  $[r, o(n^2/k^2), J]$ -diluter of the  $n$ -qubit Hamiltonian  $H_A$  with localized encoding, 0-incoherence, and energy spread  $\tilde{w} < 1/2$ , regardless of degree  $r$  or interaction strength  $J$ .*

Similar to Lemma 10, this in fact holds even if we allow inverse polynomial incoherence [1]; and like above, this seems to be a rather weak impossibility result since requiring inverse polynomial incoherence may be too strong in many situations. Can we strengthen this to rule

out dilution with constant incoherence? The proof technique in Theorem 11 does not apply for dilution, since it relies on the decay of correlation between *distant* nodes in the interaction graph of  $\tilde{H}$  (see Sec. 4.1). On the other hand, a diluter  $\tilde{H}$  can have unbounded degree, and hence constant diameter, e.g. the star graph. Nevertheless, under a computational hardness assumption, no efficient classical *algorithm* for generic constant-unfaithfulness dilution exists, even for all  $k$ -local *classical* Hamiltonians:

► **Theorem 16** (Impossibility of dilution algorithm for classical Hamiltonians). *If  $\text{coNP} \not\subseteq \text{NP/poly}$ , then for any  $\xi > 0$ ,  $\delta < 1/\sqrt{2}$ ,  $\tilde{w} \leq 1/2$ , there is no classical algorithm that given a  $k$ -local  $n$ -qubit classical Hamiltonian  $H$ , runs in  $O(\text{poly}(n))$  time to find an  $[r, O(n^{k-\xi}), J]$ -diluter of  $H$  with  $\delta$ -unfaithfulness, energy spread  $\tilde{w}$ , and any encoding  $V$  that has an  $O(n^{k-\xi})$ -bit description. This holds for any  $r$  and  $J$ .*

The above result rules out general (constructive) dilution even when the Hamiltonians are classical. For specific cases, however, dilution is possible. Our  $H_A$  (which is also a classical Hamiltonian) provides such an example, for which we can achieve dilution even with 0-unfaithfulness, in the incoherent setting:

► **Proposition 17** (0-unfaithfulness incoherent dilution and DR for  $H_A$ ). *There is a 3-local incoherent  $[2, n-1, 1]$ -diluter of  $H_A$  with 0-unfaithfulness, energy spread  $\tilde{w} = 0$ , and trivial encoding. This is also an incoherent  $[2, n-1, 1]$ -degree-reducer of  $H_A$ .*

Furthermore, combining ideas from the construction in Proposition 17 and Theorem 13, we can show that coherent dilution of  $H_A$  with polynomial interaction strength is also possible:

► **Proposition 18** (Coherent dilution and DR for  $H_A$  with polynomial interaction strength). *There is a 6-local  $[6, O(n/\epsilon^2), O(\text{poly}(n, \epsilon^{-1}))]$ -diluter of  $H_A$  with  $\epsilon$ -incoherence, energy spread  $\tilde{w} = 0$ , and trivial encoding. This is also a  $[6, O(n/\epsilon^2), O(\text{poly}(n, \epsilon^{-1}))]$ -degree-reducer of  $H_A$ .*

Note since Theorem 16 rules out constructive dilution regardless of interaction strength  $J$ , we cannot hope to prove an analogue of Theorem 13 or 14 to build coherent diluters for generic Hamiltonians, even allowing arbitrarily large interaction strength. Nevertheless, it remains an interesting open question to characterize Hamiltonians for which diluters exist, whether coherent or incoherent, with constant or large interaction strengths.

### 3.3 Connection to Quantum PCP

It might appear that our results rule out quantum degree-reduction (DR) in the context of quantum PCP (which would add to existing results [23, 5, 11, 20, 40, 6] ruling out quantum generalizations of other parts of Dinur’s PCP proof [32]). However, our results in this context (detailed in extended version[1]) currently have rather weak implications towards such a statement. The catch is that despite the apparent similarity, our gap-simulating DR is a very different notion from DR transformations used in the context of classical and quantum PCP. Gap-simulation seeks the *existence* of a Hamiltonian  $\tilde{H}$  that reproduces the properties of the groundstate(s) and *spectral gap* of an input Hamiltonian  $H$ . On the other hand, a qPCP reduction is an *algorithm* that given  $H$ , it is merely required to output some  $\tilde{H}$ , such that if the groundstate energy of  $H$  is small (or large), then so is the groundstate energy of  $\tilde{H}$ ; in other words, qPCP preserves the *promise gap*. Notice that such a  $\tilde{H}$  always *exists*, and the difficulty in qPCP reductions is to generate  $\tilde{H}$  efficiently, without knowing the groundstate energy of  $H$ . Thus, we cannot hope for an information-theoretical impossibility result (as in Theorem 11 and 15) in the qPCP setting without further restriction on the output.



To circumvent this issue, we generalize to the quantum world a natural requirement, which seems to hold in the classical world for all known PCP reductions, namely that the reduction is *constructive*: roughly, it implies a mapping on not only the CSPs (Hamiltonians) but also individual assignments (states) [16, 33]. Specifically, we require that the reduction from  $H$  to  $\tilde{H}$  preserves groundstate properties in the similar sense as gap-simulation does, and maps excited states of  $H$  (above the promise gap) to high-energy states  $\tilde{H}$ . Under this restriction, we prove that degree-reduction and dilution for quantum PCP with near-perfect coherence is impossible:

► **Theorem 19** (Limitation on qPCP-DR and qPCP-dilution (rough)). *There is no “constructive” qPCP reduction that works to degree-reduce or dilute the Hamiltonian  $H_A$  with localized encoding, if we require small incoherence  $\epsilon$  relative to the energy of the output Hamiltonian  $\tilde{H}_A$ , namely,  $\epsilon \leq o(\|\tilde{H}_A\|^{-1/2})$ .*

The proof of Theorem 19 approximately follows that of impossibility results of Lemma 10 and Theorem 15 for sparsification with close-to-perfect coherence. Unfortunately, as we explain in Sec. 4.1, strengthening these results to prove impossibility for constant error (the regime of interest for qPCP), as is done in Theorem 11, seems to require another new idea.

## 4 Proofs Overview

### 4.1 Proof Sketch for Main Theorem 11 (and related results: Theorem 15, 19 and Lemma 10)

We start with the idea underlying the impossibility of degree-reduction and dilution with (close to) perfect coherence (Lemma 10 and Theorem 15), which we refer to as “contradiction-by-energy”. For simplicity, let’s first examine the case of gap-simulation without encoding. Consider all pairs of original qubits  $(i, j)$ . The groundstates of  $H_A$  include basis states with zero or one excitations (namely, 1’s), but not 2-excitation states. Importantly, the groundstates can be obtained from the 2-excitation state by *local* operations  $\sigma_x^{(i)}$  and  $\sigma_x^{(j)}$ . Assuming the gap-simulator  $\tilde{H}_A$  of  $H_A$  does not interact the qubits  $(i, j)$ , we can express the energy of the 2-excitation state as a linear combination of the energy of 0- and 1-excitation states, up to an error of  $\mathcal{O}(\tilde{w})$  and  $\mathcal{O}(\epsilon\|\tilde{H}_A\|)$ , using the fact that we can commute  $\sigma_x^{(i)}$  and  $\sigma_x^{(j)}$  through independent parts of  $\tilde{H}_A$ . If we assume  $\tilde{w}$  is small and  $\epsilon = 0$ , the energy of the 2-excitation state cannot be distinguished from these groundstates. Thus any gap-simulator  $\tilde{H}_A$  must directly interact all pairs of qubits, which easily proves the impossibility without encoding. We can also see that if  $\epsilon > 0$ , then DR and dilution remain impossible if  $\|\tilde{H}_A\| \leq O(\epsilon^{-1})$ , e.g. when  $\epsilon$  is polynomially small. This impossibility easily extends to localized encoding, where each original qubit is encoded into  $O(1)$  qudits in the gap-simulator Hamiltonian either independently or via some constant-depth circuit. In both cases, the required  $\Omega(n)$  degree and  $\Omega(n^2)$  interaction terms implied for the non-encoded version translate to the same requirements for the encoded version up to a constant factor, proving Lemma 10 and Theorem 15.

We now explain the proof of Theorem 11 that rules out degree-reduction even with constant incoherence. Let us first consider the statement for  $H_A$  with constant  $\epsilon$  incoherence. The challenge is that the contradiction-by-energy trick used in the proof of Lemma 10 and Theorem 15 does not work for  $\epsilon = \Theta(1)$  incoherence. The problem is that the error in energy is of the order of  $\mathcal{O}(\epsilon\|\tilde{H}_A\|)$ ; this is too large for constant  $\epsilon$ , and does not allow one to distinguish the energy of ground and excited states. Instead of contradiction-by-energy, we derive a contradiction using the groundspace correlations between qubits  $(i, j)$ , where  $\epsilon$ -incoherence

only induces an error of  $\mathcal{O}(\epsilon)$ . Since  $H_A$  is gapped, then any degree-reducer Hamiltonian  $\tilde{H}_A$  of  $H_A$  must be gapped (while allowing some small energy spread  $\tilde{w}$ ) by Definition 2. We can therefore apply a result (strengthened to accommodate non-vanishing energy spread [1]) of Hastings-Koma [41] stating that groundspace correlation decays exponentially with the distance on the graph where  $\tilde{H}_A$  is embedded. Since we assume bounded degree, we can find a pair  $(i, j)$  among the original  $n$  qubits such that their supports  $(S_i, S_j)$  after a localized encoding are  $\Omega(\log n)$  distance apart, with respect to the graph metric. Hence, their correlation  $\langle V \sigma_x^{(i)} \sigma_x^{(j)} V^\dagger \rangle$  in the groundspace of  $\tilde{H}_A$  must decay as  $e^{-\Omega(\log n)} = O(1/\text{poly}(n))$ . Contradiction is achieved by the fact that for any pair of original qubits  $(i, j)$ , the groundspace of  $\tilde{H}_A$  contains a state of the form  $V(|0_i 1_j\rangle + |1_i 0_j\rangle) |0^{n-2}, \text{rest}\rangle + \mathcal{O}(\epsilon)$ , which has correlation at least  $\langle V \sigma_x^{(i)} \sigma_x^{(j)} V^\dagger \rangle = 1 - \mathcal{O}(\epsilon)$ . For sufficiently small  $\epsilon$  and  $\tilde{w}$ , this constant correlation from the latter lower bound contradicts the  $O(1/\text{poly}(n))$  upper bound from the Hastings-Koma result.

The second part of Theorem 11 proves impossibility of incoherent DR for  $H_B$  with  $\delta$ -unfaithfulness. Since  $H_B$  has a unique groundstate that can be shown to have constant correlation between any pair of original qubits  $(i, j)$ , we can apply the same argument above for  $H_A$  and show a contradiction with the Hastings-Koma's vanishing upper bound of  $O(1/\text{poly}(n))$  for small  $\delta$  and  $\tilde{w}$ .

We now remark how these impossibility proofs can be extended to the context of quantum PCP. The contradiction-by-energy idea in Lemma 10 and Theorem 15 can indeed be generalized in this context. Under a reasonable restriction on the reduction – namely that the energy of non-satisfying assignments (frustrated or excited states) after the mapping is lower bounded by the promise gap – degree-reduction or dilution for quantum PCP is not generally possible with close-to-perfect (namely inverse polynomial) coherence (Theorem 19). However, this impossibility proof would not work when constant incoherence is allowed. To move to contradiction-by-correlation as in Theorem 11, we need to use some form of Hastings-Koma, which requires a spectral gap in  $\tilde{H}$ . Thus, more innovation is needed as it may be an unnecessarily strong requirement for quantum PCP to preserve the spectral gap.

## 4.2 Overview of Remaining Proofs

**Proof sketch: Equivalence between coherent and incoherent gap-simulations for unique groundstates (Lemma 4).** We want to show that incoherent gap-simulation implies coherent gap-simulation, in the case of unique groundstate of the original Hamiltonian  $H$ . A naïve approach using the small error per groundstate of the gap-simulator will not work due to possible degeneracy in the groundspace of the simulator  $\tilde{H}$ ; this (possibly exponential) degeneracy could add an unwanted exponential factor. Hence, we explicitly construct the subspace on which the ancilla qubits should be projected by  $P_{\text{anc}}$ . The main observation is that since faithful gap-simulation implies that any state in the groundspace of  $\tilde{H}$  must be close to the space spanned by this  $P_{\text{anc}}$ , the dimensions of  $P_{\text{anc}}$  and the groundspace of  $\tilde{H}$  must be the same. A sequence of simple arguments then allows us to derive a bound on the incoherence of any state (i.e., its norm after the incoherence operator in Eq. (3) is applied).

**Proof sketch: DR of any classical Hamiltonian (Proposition 12).** Here we follow the standard classical DR (as in [32]) in which each variable (of degree  $d$ ) is replaced by  $d$  variables, and a ring of equality constraints on these variables is added to ensure they are the same. The proof that this satisfies our gap-simulation definition is straightforward.

**Proof sketch: Coherent DR of any Hamiltonian with  $\Omega(1/\text{poly}(n))$  spectral gap using polynomial interaction strength (Theorem 13).**

The construction is based on mapping the quantum phase estimation (PE) circuit [53] to a Hamiltonian, using a modified version of Kitaev’s circuit-to-Hamiltonian construction [47]. The PE circuit can write the energy of any eigenstate of a given  $H$  in an ancilla register, up to polynomial precision using polynomial overhead. The degree of the Hamiltonian is reduced by “sparsifying” the circuit before converting to the Hamiltonian. To repair the incoherence due to different histories, we run the circuit backwards, removing entanglement between the ancilla and the original register. To achieve  $\epsilon$ -incoherence, we add  $O(\text{poly}(n)/\epsilon^2)$  identity gates to the end of the circuit. The eigenvalue structure of the original Hamiltonian  $H$  is restored by imposing energy penalties on the energy bit-string written on the ancilla by the PE circuit. This yields a full-spectrum simulation of  $H$ , which also implies a gap-simulation of  $H$ .

**Proof sketch: Impossibility of generic dilution algorithm (Theorem 16).**

Ref. [29] shows that under the assumption  $\text{coNP} \not\subseteq \text{NP/poly}$ , there is no poly-time algorithm to “compress” vertex-cover problems on  $n$ -vertex  $k$ -uniform hypergraphs and decide the problem by communicating  $O(n^{k-\xi})$  bits for any  $\xi > 0$  to a computationally unbounded oracle. Suppose towards a contradiction that  $\mathcal{A}$  is a poly-time algorithm to dilute any  $k$ -local classical Hamiltonian; we use it to derive a compression algorithm for vertex cover. To this end,  $\mathcal{A}$  is given a classical  $k$ -local Hamiltonian  $H$  encoding a vertex cover problem;  $\mathcal{A}$  produces the diluter  $\tilde{H}$  with  $O(n^{k-2\xi})$  terms and some encoding  $V$  described by  $O(n^{k-2\xi})$  bits. Using Green’s function perturbation theory, we show that  $\tilde{H}$  can be written using only  $\log(n)$ -bit precision as  $\tilde{H}'$  with  $O(1)$  error in the quasi-groundspace (even accounting for degeneracy). We then communicate  $(\tilde{H}', V)$  to the oracle by sending  $O(n^{k-2\xi} \log n) = O(n^{k-\xi})$  bits. The oracle then uses any groundstate of  $\tilde{H}'$ , which has large overlap with groundstates of  $H$  for small  $\delta$  and high precision, to decide the vertex cover problem and transmit back the answer.

**Proof sketch: Incoherent dilution and DR of  $H_A$  (Proposition 17).**

We use here the usual translation of a classical circuit to a CSP:  $n - 1$  qubits in a tree structure are used to simulate counting of the number of 1s among the original qubits, and the CSP checks the correctness of this (local) computation. The “history” of the computation is written on the ancilla qubits. Since different strings have different such histories, the construction is incoherent.

**Proof sketch: Coherent dilution and DR of  $H_A$  with polynomial interaction strength (Proposition 18).**

We improve upon the construction in Prop. 17 and Theorem 13 to obtain a coherent diluter of  $H_A$  with polynomial interaction strength. The key is an  $O(n)$ -length circuit similar to that of Prop. 17 with a circuit that counts the number of 1s in the same tree geometry. Using the same tricks in Theorem 13 to uncompute computational histories and add identity gates at the end, we show that this leads to a coherent gap-simulator of  $H_A$  with  $\epsilon$ -incoherence and  $O(n/\epsilon^2)$  terms.

**Proof sketch: Coherent DR for any Hamiltonian using exponential or larger interaction strength (Theorem 14).**

In order to provide generic coherent degree-reduction for any local Hamiltonian without restriction on the spectral gap  $\gamma$ , we first show that perturbative gadgets [45, 54] can be used for gap-simulation. The proofs make use of Green’s function machinery to bound incoherence  $\epsilon$ , which can be made small after every application of the gadgets using interactions of strength  $O(\text{poly}(n)/(\gamma\epsilon)^{O(1)})$ . This allows us to construct a

degree-reducer for any  $k$ -local Hamiltonian by a sequence of perturbative gadget applications. In the first part of the sequence, we reduce the locality of individual Hamiltonian terms to 3-local via  $O(\log k)$  serial applications of subdivision gadgets [54], and each 3-local term is further reduced to 2-local via “3-to-2-local” gadgets [54]. Then, each original qubit is isolated from each other by subdivision gadgets so that they only interact with  $O(n^{k-1})$  ancilla qubits that mediate interactions. Finally, applying fork gadgets [54] in  $O(\log n)$  iterations allows us to reduce maximum degree of these original qubits to 6, generating our desired degree-reducer. It is this last part that causes the exponential blow-up in the interaction strength relative to  $\gamma\epsilon$ , so as to maintain the gap-simulation.

**Proof sketch: Generalized Hastings-Koma.** In Ref. [41], Hastings and Koma proved the exponential decay of correlations in the quasi-groundspace of a Hamiltonian  $H$  consisting of finite-range (or exponentially decaying) interactions between particles embedded on a graph. They assume that the system is spectrally gapped, and has vanishing energy spread as the system size  $n \rightarrow \infty$ . Their proof is based on the relationship between the correlation  $\langle \sigma^{(i)} \sigma^{(j)} \rangle$  they want to upper bound, and the commutator  $\langle [e^{-iHt} \sigma^{(i)} e^{iHt}, \sigma^{(j)}] \rangle$ . By applying the Lieb-Robinson bound [51] on the latter, and integrating out the time  $t$ , they show that under the above conditions, the correlations between operators acting on particles  $i$  and  $j$  decay exponentially with the graph-theoretic distance between the particles. For application to the gap-simulation framework, we need to generalize their result to cases where the energy spread is not assumed to vanish with the system size. This is done by a careful modification of their proofs where we optimize the bounds and integration parameters so that errors due to the non-zero energy spread are suppressed.

## 5 Discussion and outlook

We have initiated the rigorous research of resources required for analog simulations of Hamiltonians, and proved unexpected impossibility results for Hamiltonian sparsification. Instead of working with full-spectrum simulations [22, 27], we use a new, relaxed definition of gap-simulation that is motivated by minimal requirements in physics. We note that impossibility results proven in a relaxed framework are of course stronger.

It would be very interesting to improve our understanding of the new framework of gap-simulations presented here, and clarify its applicability. As a start, it will be illuminating to find applications of gap-simulations in cases where full-spectrum simulations as in Ref. [22, 27] are unknown or difficult to achieve. For example, our Prop. 18 achieves dilution of  $H_A$  with gap-simulation, but we do not know how to do so with full-spectrum simulation. Such simulations can enable experimental studies of these physical systems, by reducing resources required for analog simulations. Moreover, in many-body quantum physics, tools to construct “equivalent” Hamiltonians that preserve groundstate properties are of great utility. In this context, the study of gap-simulations can potentially lead to better understanding of universal behaviors in quantum phases of matter, which are characterized only by groundstate physics [56]. Another possible application of gap-simulators may be in the design of Hamiltonian-based quantum algorithms. In adiabatic algorithms [35], it is well known that the higher parts of the spectrum of the final and initial Hamiltonians can significantly affect the adiabatic gap [34, 31, 13]; gap-simulating these final and initial Hamiltonians by others will not affect the final groundstate, and can sometimes dramatically improve on the gap along the adiabatic path. Gap-simulations may also be a useful tool for tailoring the Hamiltonians used in other Hamiltonian-based algorithms such as QAOA [36].

We note that incoherent but faithful gap-simulations can be very interesting despite the apparent violation of the quantum requirement for coherence. For example, in adiabatic algorithms [35], we only want to arrive at one of the solutions (groundstates) to a quantum constraint satisfaction problem. In addition, in quantum NP [8], one is interested only in whether a certain eigenvalue *exists*, and not in the preservation of the entire groundspace. However, in the context of quantum simulation and many-body physics, maintaining coherence seems to be crucial for transporting all the physical properties of the groundspace. One would also expect maintaining coherence to be important when gap-simulating a subsystem (perhaps in an unknown state) of a larger system.

We remark that our framework deliberately avoids requiring that the eigenvalue structure of the spectrum be maintained even in its low-lying part, so as to provide a minimal but still interesting definition. Indeed, when simulating the groundspace, or a quasi-groundspace with small energy spread, this structure is not important. Nevertheless, one can imagine an intermediate definition, in which full-spectrum simulation is too strong, but the structure of a significant portion of the lower part of the spectrum matters. It might be interesting to extend the framework of gap-simulations to allow for such intermediate cases in which, for example, Gibbs states at low (but not extremely low) temperatures are faithfully simulated.

A plethora of open questions arise in the context of sparsification. First, it will be very interesting to find more examples where degree-reduction and/or dilution are possible, or are helpful from the perspective of physical implementations. Assuming bounded interaction strength, which is generally a limitation of physical systems, can we rigorously characterize which Hamiltonians can be coherently (or incoherently) degree-reduced? Of course, similar questions can be asked about dilution. It will also be interesting to consider saving other resources such as the dimensionality of the particles, which would be a generalization of alphabet-reductions from the context of PCP to Hamiltonian sparsification.

Our results on the impossibility of dilution are weaker than those for DR. Can we strengthen these to stronger information-theoretical results, by finding a quantum Hamiltonian for whom a diluter does not *exist* with constant incoherence, or even constant unfaithfulness?

We mention here that the classical graph sparsification results of Ref. [14, 15] can be viewed as dilution of a graph while approximately maintaining its spectrum. These results have been generalized to the matrix setting in Ref. [58]; however, this generalization does not seem to be useful in the context of diluting the interaction graph of a local Hamiltonian. The result of Ref. [58] shows that for sums of  $d \times d$  positive Hermitian matrices,  $O(d)$  matrices are sufficient to reproduce the spectral properties to good approximation, improving over Chernoff-like bounds [10]. While this in principle allows one to approximate a sum of terms by a sum of fewer terms, the required number of terms grows as  $d = 2^{\Omega(n)}$  for quantum Hamiltonians on  $n$  qubits, and is thus irrelevant in our context.

Improving the *geometry* of the simulators is another important task that is relevant for applications of Hamiltonian sparsification to physical implementations. Ref. [49] has devised a method of converting the NP-complete Ising model Hamiltonian ( $H = \sum_{ij} J_{ij} \sigma_z^{(i)} \sigma_z^{(j)} + \sum_i h_i \sigma_z^{(i)}$ ) on  $n$  qubits to a new Hamiltonian on  $O(n^2)$  qubits with interactions embedded on a 2D lattice, and sharing the same low-energy spectrum. Their construction encodes each edge  $\sigma_z^{(i)} \sigma_z^{(j)}$  as a new qubit, and corresponds to an incoherent degree-reducer, where the new groundstates are non-locally encoded version of the original states. Our Prop. 12 also provides incoherent DR of these Hamiltonians, and without encoding, but the geometry is not in 2D; it will be interesting to improve our Prop. 12 as well as our other positive Theorems 13 and 14 to hold using a spatially local  $\tilde{H}$ . We note that if we allow the overhead of polynomial interaction strength, then it should be straightforward to extend the circuit-to-Hamiltonian

construction in Theorem 13 for analog simulations of local Hamiltonians on a 2D lattice, by ordering the gates in a snake-like fashion on the lattice similar to Ref. [54, 9]. Identifying situations where DR in 2D with bounded interaction strength is possible remains an open question.

A different take on the geometry question is to seek gap-simulators which use a single (or few) ancilla qubits that strongly interact with the rest. This may be relevant for physical systems such as neutral atoms with Rydberg blockade [57], where an atom in a highly excited level may have a much larger interaction radius, while no two atoms can be excited in each other's vicinity.

Can we improve our results about quantum PCP, and show impossibility of qPCP-DR with constant incoherence? This would make our impossibility results interesting also in the qPCP context, as they would imply impossibility of DR in the qPCP regime of constant error, under a rather natural restriction on the qPCP reduction. This would complement existing impossibility results on various avenues towards qPCP [23, 5, 11, 20, 40, 6, 3]. Nevertheless, it seems that proving such a result might require a significantly further extension of Hastings-Koma, which may be of interest on its own.

Finally, we mention a possibly interesting variant of gap-simulation, which we call *weak* gap-simulation (see details in extended version[1]). Here, the groundspace is simulated in an *excited* eigenspace of the simulating Hamiltonian, spectrally gapped from above and below, rather than in its groundspace. This can be useful in the context of Floquet Hamiltonian engineering in periodically driven quantum systems, where eigenvalues are meaningful only up to a period, and thus a spectral gap in the middle of the spectrum is analogous to a spectral gap above the groundspace [39]. We are able to show how to weakly gap-simulate  $H_A$  to provide dilution with *constant* incoherence and *bounded* interaction strength – a task which we currently do not know how to do using “standard” gap-simulation. It remains open whether one can show stronger possibility results under weak gap-simulation. If not, can the impossibility results presented here be extended to the weak-gap-simulation setting? This might require an even stronger extension of Hastings-Koma's theorem.

Overall, we hope that the framework, tools, and results presented here will lead to progress in understanding the possibilities and limitations in simulating Hamiltonians by other Hamiltonians – an idea that brings the notion of *reduction* from classical computer science into the quantum realm, and constitutes one of the most important contributions of the field of quantum computational complexity to physics.

---

## References

- 1 See extended version at [arXiv:1804.11084](https://arxiv.org/abs/1804.11084).
- 2 D. Aharonov, I. Arad, and S. Irani. Efficient algorithm for approximating one-dimensional ground states. *Phys. Rev. A*, 82:012315, 2010.
- 3 D. Aharonov, I. Arad, and T. Vidick. The Quantum PCP Conjecture. *ACM SIGACT News*, 44:47–79, 2013.
- 4 D. Aharonov and M. Ben-Or. Fault-tolerant Quantum Computation with Constant Error. In *Proceedings of the 29th ACM Symposium on Theory of Computing, STOC '97*, pages 176–188, 1997.
- 5 D. Aharonov and L. Eldar. On the Complexity of Commuting Local Hamiltonians, and Tight Conditions for Topological Order in Such Systems. In *Proceedings of the 52nd Symposium on Foundations of Computer Science, FOCS '11*, pages 334–343, 2011.

- 6 D. Aharonov and L. Eldar. The Commuting Local Hamiltonian Problem on Locally Expanding Graphs is Approximable in NP. *Quantum Information Processing*, 14(1):83–101, 2015.
- 7 D. Aharonov, A. W. Harrow, Z. Landau, D. Nagaj, M. Szegedy, and U. Vazirani. Local Tests of Global Entanglement and a Counterexample to the Generalized Area Law. In *Proceedings of the 55th Symposium on Foundations of Computer Science*, FOCS '14, pages 246–255, 2014.
- 8 D. Aharonov and T. Naveh. Quantum NP - A Survey. *CoRR*, 2002. [arXiv:quant-ph/0210077](#).
- 9 D. Aharonov, W. van Dam, J. Kempe, Z. Landau, S. Lloyd, and O. Regev. Adiabatic Quantum Computation is Equivalent to Standard Quantum Computation. *SIAM J. Comput.*, 37(1):166–194, 2007.
- 10 R. Ahlswede and A. Winter. Strong converse for identification via quantum channels. *IEEE Transactions on Information Theory*, 48(3):569–579, 2002.
- 11 I. Arad. A Note About a Partial No-go Theorem for Quantum PCP. *Quantum Info. Comput.*, 11(11-12):1019–1027, 2011.
- 12 A. Aspuru-Guzik and P. Walther. Photonic quantum simulators. *Nature Physics*, 8(4):285–291, 2012.
- 13 Y. Atia and D. Aharonov, 2018. in preparation.
- 14 J. Batson, D. A. Spielman, and N. Srivastava. Twice-Ramanujan Sparsifiers. *SIAM J. Comput.*, 41:1704–1721, 2012.
- 15 J. Batson, D. A. Spielman, N. Srivastava, and S.-H. Teng. Spectral Sparsification of Graphs. *Commun. ACM*, 56:87–94, 2013.
- 16 E. Ben-Sasson, O. Goldreich, P. Harsha, M. Sudan, and S. Vadhan. Robust PCPs of Proximity, Shorter PCPs, and Applications to Coding. *SIAM J. Comput.*, 36(4):889–974, 2006.
- 17 H. Bernien et al. Probing many-body dynamics on a 51-atom quantum simulator. *Nature*, 551(7682):579–584, 2017.
- 18 R. Blatt and C. F. Roos. Quantum simulations with trapped ions. *Nature Physics*, 8(4):277–284, 2012.
- 19 I. Bloch, J. Dalibard, and S. Nascimbène. Quantum simulations with ultracold quantum gases. *Nature Physics*, 8(4):267–276, 2012.
- 20 F.G.S.L. Brandão and A. W. Harrow. Product-state Approximations to Quantum Ground States. *Comm. Math. Phys.*, 342:47–80, 2016.
- 21 S. Bravyi, D. P. DiVincenzo, D. Loss, and B. M. Terhal. Quantum Simulation of Many-Body Hamiltonians Using Perturbation Theory with Bounded-Strength Interactions. *Phys. Rev. Lett.*, 101:070503, 2008.
- 22 S. Bravyi and M. Hastings. On complexity of the quantum Ising model. *CoRR*, 2014. [arXiv:1410.0703](#).
- 23 S. Bravyi and M. Vyalyi. Commutative Version of the Local Hamiltonian Problem and Common Eigenspace Problem. *Quantum Info. Comput.*, 5(3):187–215, 2005.
- 24 Y. Cao, R. Babbush, J. Biamonte, and S. Kais. Hamiltonian gadgets with reduced resource requirements. *Phys. Rev. A*, 91:012315, 2015.
- 25 Y. Cao and D. Nagaj. Perturbative gadgets without strong interactions. *Quantum Inf. Comput.*, 15:1197–1222, 2015.
- 26 J. I. Cirac and P. Zoller. Goals and opportunities in quantum simulation. *Nature Physics*, 8(4):264–266, 2012.
- 27 T. Cubitt, A. Montanaro, and S. Piddock. Universal Quantum Hamiltonians. *CoRR*, 2017. [arXiv:1701.05182](#).



- 28 T. S. Cubitt, D. Perez-Garcia, and M. M. Wolf. Undecidability of the spectral gap. *Nature*, 528(7581):207–211, 2015.
- 29 H. Dell and D. van Melkebeek. Satisfiability Allows No Nontrivial Sparsification Unless the Polynomial-time Hierarchy Collapses. In *Proceedings of the 42nd ACM Symposium on Theory of Computing*, STOC '10, pages 251–260. ACM, 2010.
- 30 R. H. Dicke. Coherence in Spontaneous Radiation Processes. *Phys. Rev.*, 93:99–110, 1954.
- 31 N. G. Dickson and M. H. Amin. Algorithmic approach to adiabatic quantum optimization. *Phys. Rev. A*, 85:032303, 2012.
- 32 I. Dinur. The PCP Theorem by Gap Amplification. *J. ACM*, 54(3), 2007.
- 33 I. Dinur, O. Goldreich, and T. Gur. Every set in  $\mathcal{P}$  is strongly testable under a suitable encoding. *Electronic Colloquium on Computational Complexity (ECCC)*, 25:50, 2018.
- 34 E. Farhi, J. Goldstone, S. Gutman, and D. Nagaj. How to Make the Quantum Adiabatic Algorithm Fail. *Int. J. Quantum Inf.*, 6:503–516, 2008.
- 35 E. Farhi, J. Goldstone, S. Gutmann, and M. Sipser. Quantum Computation by Adiabatic Evolution. *CoRR*, 2000. [arXiv:quant-ph/0001106](https://arxiv.org/abs/quant-ph/0001106).
- 36 E. Farhi and A. W Harrow. Quantum Supremacy through the Quantum Approximate Optimization Algorithm. *CoRR*, 2016. [arXiv:1602.07674](https://arxiv.org/abs/1602.07674).
- 37 R. P. Feynman. Simulating Physics with Computers. *International Journal of Theoretical Physics*, 21(6), 1982.
- 38 I. M. Georgescu, S. Ashhab, and F. Nori. Quantum simulation. *Rev. Mod. Phys.*, 86:153–185, 2014.
- 39 M. Grifoni and P. Hänggi. Driven quantum tunneling. *Physics Reports*, 304(5-6):229–354, 1998.
- 40 M. B. Hastings. Trivial Low Energy States for Commuting Hamiltonians, and the Quantum PCP Conjecture. *Quantum Info. Comput.*, 13(5-6):393–429, 2013.
- 41 M. B. Hastings and T. Koma. Spectral Gap and Exponential Decay of Correlations. *Comm. Math. Phys.*, 265:781–804, 2006.
- 42 A. A. Houck, H. E. Türeci, and J. Koch. On-chip quantum simulation with superconducting circuits. *Nature Physics*, 8(4):292–299, 2012.
- 43 S. P. Jordan and E. Farhi. Perturbative gadgets at arbitrary orders. *Phys. Rev. A*, 77:062329, 2008.
- 44 A. Keesling et al. Probing quantum critical dynamics on a programmable Rydberg simulator. *CoRR*, 2018. [arXiv:1809.05540](https://arxiv.org/abs/1809.05540).
- 45 J. Kempe, A. Kitaev, and O. Regev. The Complexity of the Local Hamiltonian Problem. *SIAM J. Comput.*, 35:1070–1097, 2006.
- 46 A. Y. Kitaev. Fault-tolerant quantum computation by anyons. *Annals of Physics*, 303(1):2–30, 2003.
- 47 A. Y. Kitaev, A. Shen, and M. N. Vyalyi. *Classical and Quantum Computation*. American Mathematical Society, 2002.
- 48 E. Knill, R. Laflamme, and W. H. Zurek. Resilient quantum computation: error models and thresholds. *Proc. R. Soc. London, Ser. A*, 454(1969):365–384, 1998.
- 49 W. Lechner, P. Hauke, and P. Zoller. A quantum annealing architecture with all-to-all connectivity from local interactions. *Science Advances*, 1(9), 2015.
- 50 M. Levin and X.-G. Wen. Detecting Topological Order in a Ground State Wave Function. *Phys. Rev. Lett.*, 96:110405, 2006.
- 51 E. H. Lieb and D. W. Robinson. The finite group velocity of quantum spin systems. *Comm. Math. Phys.*, 28(3):251–257, 1972.
- 52 S. Lloyd. Universal Quantum Simulators. *Science*, 273(5278):1073–8, 1996.
- 53 M. A. Nielsen and I. L. Chuang. *Quantum Computation and Quantum Information*. Cambridge University Press, 10th edition, 2011.



- 54 R. Oliveira and B. M. Terhal. The complexity of quantum spin systems on a two-dimensional square lattice. *Quantum Inf. Comput.*, 8:900–924, 2008.
- 55 J. Preskill. Quantum Computing in the NISQ era and beyond. *Quantum*, 2:79, 2018.
- 56 S. Sachdev. *Quantum phase transitions*. Cambridge University Press, 2011.
- 57 M. Saffman, T. G. Walker, and K. Mølmer. Quantum information with Rydberg atoms. *Rev. Mod. Phys.*, 82:2313–2363, 2010.
- 58 M. K. de Carli Silva, N. J. A. Harvey, and C. M. Sato. Sparse Sums of Positive Semidefinite Matrices. *ACM Trans. Algorithms*, 12(1):9:1–9:17, 2015.
- 59 J. Simon, W. S. Bakr, R. Ma, M. Eric Tai, P. M. Preiss, and M. Greiner. Quantum simulation of antiferromagnetic spin chains in an optical lattice. *Nature*, 472(7343):307–312, 2011.
- 60 D. A. Spielman and N. Srivastava. Graph Sparsification by Effective Resistances. *SIAM J. Comput.*, 40(6):1913–1926, 2011.
- 61 D. A. Spielman and S.-H. Teng. Nearly-Linear Time Algorithms for Preconditioning and Solving Symmetric, Diagonally Dominant Linear Systems. *CoRR*, 2006. [arXiv:0607105](https://arxiv.org/abs/0607105).
- 62 D. A. Spielman and S.-H. Teng. Spectral Sparsification of Graphs. *SIAM J. Comput.*, 40(4):981–1025, 2011.
- 63 X. G. Wen. Topological order in rigid states. *Int. J. Mod. Phys. B*, 04(02):239–271, 1990.
- 64 J. Zhang et al. Observation of a many-body dynamical phase transition with a 53-qubit quantum simulator. *Nature*, 551(7682):601–604, 2017.



# On Solving Linear Systems in Sublinear Time

Alexandr Andoni<sup>1</sup>

Columbia University, New York, NY, USA  
andoni@cs.columbia.edu

Robert Krauthgamer<sup>2</sup>

Weizmann Institute of Science, Rehovot, Israel  
robert.krauthgamer@weizmann.ac.il

Yosef Pogrow

Weizmann Institute of Science, Rehovot, Israel  
yosef.pogrow@weizmann.ac.il

---

## Abstract

We study *sublinear* algorithms that solve linear systems locally. In the classical version of this problem the input is a matrix  $S \in \mathbb{R}^{n \times n}$  and a vector  $b \in \mathbb{R}^n$  in the range of  $S$ , and the goal is to output  $x \in \mathbb{R}^n$  satisfying  $Sx = b$ . For the case when the matrix  $S$  is symmetric diagonally dominant (SDD), the breakthrough algorithm of Spielman and Teng [STOC 2004] approximately solves this problem in near-linear time (in the input size which is the number of non-zeros in  $S$ ), and subsequent papers have further simplified, improved, and generalized the algorithms for this setting.

Here we focus on computing one (or a few) coordinates of  $x$ , which potentially allows for sublinear algorithms. Formally, given an index  $u \in [n]$  together with  $S$  and  $b$  as above, the goal is to output an approximation  $\hat{x}_u$  for  $x_u^*$ , where  $x^*$  is a fixed solution to  $Sx = b$ .

Our results show that there is a qualitative gap between SDD matrices and the more general class of positive semidefinite (PSD) matrices. For SDD matrices, we develop an algorithm that approximates a single coordinate  $x_u$  in time that is polylogarithmic in  $n$ , provided that  $S$  is sparse and has a small condition number (e.g., Laplacian of an expander graph). The approximation guarantee is additive  $|\hat{x}_u - x_u^*| \leq \epsilon \|x^*\|_\infty$  for accuracy parameter  $\epsilon > 0$ . We further prove that the condition-number assumption is necessary and tight.

In contrast to the SDD matrices, we prove that for certain PSD matrices  $S$ , the running time must be at least polynomial in  $n$  (for the same additive approximation), even if  $S$  has bounded sparsity and condition number.

**2012 ACM Subject Classification** Theory of computation → Streaming, sublinear and near linear time algorithms

**Keywords and phrases** Linear systems, Laplacian solver, Sublinear time, Randomized linear algebra

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2019.3

**Related Version** Full version at [arXiv:1809.02995](https://arxiv.org/abs/1809.02995).

**Acknowledgements** The authors thank anonymous reviewers for suggesting additional relevant references.

---

<sup>1</sup> Work supported in part by Simons Foundation (#491119), NSF grants CCF-1617955 and CCF-1740833.

<sup>2</sup> Work supported in part by ONR Award N00014-18-1-2364, the Israel Science Foundation grant #1086/18, a Minerva Foundation grant, and a Google Faculty Research Award.



## 1 Introduction

Solving linear systems is a fundamental problem in many areas. A basic version of the problem has as input a matrix  $A \in \mathbb{R}^{n \times n}$  and a vector  $b \in \mathbb{R}^n$ , and the goal is to find  $x \in \mathbb{R}^n$  such that  $Ax = b$ . The fastest known algorithm for general  $A$  is by a reduction to matrix multiplication, and takes  $O(n^\omega)$  time, where  $\omega < 2.373$  [19] is the matrix multiplication exponent. When  $A$  is sparse, one can do better (by applying the conjugate gradient method to the equivalent positive semidefinite (PSD) system  $A^\top Ax = A^\top b$ , see for example [32]), namely,  $O(mn)$  time where  $m = \text{nnz}(A)$  is the number of non-zeros in  $A$ . This  $O(mn)$  bound for exact solvers assumes exact arithmetic, and in practice, one seeks fast approximate solvers.

One interesting subclass of PSD matrices is that of symmetric diagonally dominant (SDD) matrices.<sup>3</sup> Many applications require solving linear systems in SDD matrices, and most notably their subclass of graph-Laplacian matrices, see e.g. [32, 36, 14]. Solving SDD linear systems received a lot of attention in the past decade after the breakthrough result by Spielman and Teng in 2004 [31], showing that a linear system in SDD matrix  $S$  can be solved approximately in near-linear time  $O(m \log^{O(1)} n \log \frac{1}{\epsilon})$ , where  $m = \text{nnz}(S)$  and  $\epsilon > 0$  is an accuracy parameter. A series of improvements led to the state-of-the-art SDD solver of Cohen et al. [14] that runs in near-linear time  $O(m \sqrt{\log n} (\log \log n)^{O(1)} \log \frac{1}{\epsilon})$ . Recent improvements extend to connection Laplacians [24]. Obtaining similar results for all PSD matrices remains a major open question.

Motivated by fast linear-system solvers in alternative models, here we study which linear systems can be solved in *sublinear time*. We can hope for such sublinear times if only one (or a few) coordinates of the solution  $x \in \mathbb{R}^n$  are sought. Formally, given a matrix  $S \in \mathbb{R}^{n \times n}$ , a vector  $b \in \mathbb{R}^n$ , and an index  $u \in [n]$ , we want to approximate the coordinate  $x_u$  of a solution  $x \in \mathbb{R}^n$  to the linear system  $Sx = b$  (assume for now the solution is unique), and we want the running time to be *sublinear in  $n$* .

Our main contribution is a *qualitative separation* between the class of SDD matrices and the larger class of PSD matrices, as follows. For well-conditioned SDD matrices  $S$ , we develop a (randomized) algorithm that approximates a single coordinate  $x_u$  fast – in  $\text{polylog}(n)$  time. In contrast, for some well-conditioned PSD (but not SDD) matrices  $S$ , we show that the same task requires  $n^{\Omega(1)}$  time. In addition, we justify the dependence on the condition number.

Our study is partly motivated by the advent of *quantum* algorithms that solve linear systems in sublinear time, which were introduced in [20], and subsequently improved in [2, 11], and meanwhile used for a number of (quantum) machine learning algorithms (see, e.g., the survey [15]). In particular, the model in [20] considers a system  $Ax = b$  given: (1) oracle access to entries of  $A$  (including fast access to the  $j$ -th non-zero entry in the  $i$ -th row), and (2) a fast black-box procedure to prepare a quantum state  $|b\rangle = \sum_i \frac{b_i|i\rangle}{\|\sum_i b_i|i\rangle\|}$ . Then, if the matrix  $A$  has condition number  $\kappa$ , at most  $d$  non-zeros per row/column, and  $\|A\| = 1$ , their quantum algorithm runs in time  $\text{poly}(\kappa, d, 1/\epsilon)$ , and outputs a quantum state  $|\hat{x}\rangle$  within  $\ell_2$ -distance  $\epsilon$  from  $|x\rangle = \frac{\sum_i x_i|i\rangle}{\|\sum_i x_i|i\rangle\|}$ . The runtime was later improved in [11] to depend logarithmically on  $1/\epsilon$ . (The original goal of [20] was different – to output a “classical” value, a linear combination of  $|x\rangle$  – and for this goal the improved dependence on  $1/\epsilon$  is not possible unless  $BQP = PP$ .) These quantum sublinear-time algorithms raise the question whether there are analogous classical algorithms for the same problems; for example, a very recent success story is a classical algorithm [35] for a certain variant of recommendation

<sup>3</sup> A symmetric matrix  $S \in \mathbb{R}^{n \times n}$  is called SDD if  $S_{ii} \geq \sum_{j \neq i} |S_{ij}|$  for all  $i \in [n]$ .

systems, inspired by an earlier quantum algorithm [22]. Our lower bound precludes a classical analogue to the aforementioned linear-system solver, which works for all matrices  $A$  and in particular for PSD ones.

**Problem Formulation.** To formalize the problem, we need to address a common issue for linear systems – they may be underdetermined and thus have many solutions  $x$ , which is a nuisance when solving for a single coordinate. We require that the algorithm approximates a single solution  $x^*$ , in the sense that invoking the algorithm with different indices  $u \in [n]$  will output coordinates that are all consistent with one “global” solution. This formulation follows the concept of Local Computation Algorithms, see Section 1.3.

Our formal requirement is thus as follows. Given a matrix  $S \in \mathbb{R}^{n \times n}$ , a vector  $b \in \mathbb{R}^n$  in the range (column space) of  $S$ , and an accuracy parameter  $\epsilon > 0$ , there exists  $x^* \in \mathbb{R}^n$  satisfying  $Sx^* = b$ , such that upon query  $u \in [n]$  the (randomized) algorithm outputs  $\hat{x}_u$  that satisfies

$$\forall u \in [n], \quad \Pr \left[ |\hat{x}_u - x_u^*| \leq \epsilon \|x^*\|_\infty \right] \geq \frac{3}{4}. \quad (1)$$

This guarantee corresponds (modulo amplification of the success probability) to reporting a solution  $\hat{x} \in \mathbb{R}^n$  with  $\|\hat{x} - x^*\|_\infty \leq \epsilon \|x^*\|_\infty$ . We remark that the guarantee in [31] is different, that  $\|\hat{x} - x^*\|_S \leq \epsilon \|x^*\|_S$  where  $\|y\|_S \stackrel{\text{def}}{=} \sqrt{y^\top S y}$ , see also Section 1.4.

**Basic Notation.** Given a (possibly edge-weighted) undirected graph  $G = (V, E)$ , we assume for convenience  $V = [n]$ . Its Laplacian is the matrix  $L_G \stackrel{\text{def}}{=} D - A \in \mathbb{R}^{n \times n}$ , where  $A$  is the (weighted) adjacency matrix of  $G$ , and  $D$  is the diagonal matrix of (weighted) degrees in  $G$ . It is well-known that all Laplacians are SDD matrices, which in turn are always PSD.

The *sparsity* of a matrix is the maximum number of non-zero entries in a single row/column. The *condition number* of a PSD matrix  $S$ , denoted  $\kappa(S)$ , is the ratio between its largest and smallest non-zero eigenvalues.<sup>4</sup> For example, for the Laplacian  $L_G$  of a connected  $d$ -regular graph  $G$ , let  $\mu_1 \leq \dots \leq \mu_n$  denote its eigenvalues, then the condition number is  $\kappa(L_G) = \Theta(\frac{d}{\mu_2})$ . This follows from two well-known facts, that  $\mu_n \in [d, 2d]$ , and that  $\mu_2 > \mu_1 = 0$  if  $G$  is connected ( $\mu_2$  is called the spectral gap). Throughout,  $\|A\|$  denotes the spectral norm of a matrix  $A$ , and  $A^+$  denotes the Moore-Penrose pseudo-inverse of  $A$ .<sup>5</sup>

## 1.1 Our Results

Below we describe our results, which include both algorithms and lower bounds. First, we present a polylogarithmic-time algorithm for the simpler case of Laplacian matrices, and then we generalize it to all SDD matrices. We further prove two lower bounds, which show that our algorithms cannot be substantially improved to handle more general inputs or to run faster. The first lower bound shows that general PSD matrices require polynomial time, thereby showing a strong separation from the SDD case. The second one shows that our SDD algorithm’s dependence on the condition number is necessary and in fact near-tight.

<sup>4</sup> Our definition is in line with the standard one, for a general matrix  $A$ , which uses singular values instead of eigenvalues. If  $A$  is singular, one could alternatively define  $\kappa(A) = \infty$ , which would only make the problem easier (say to bound performance in terms of  $\kappa$ ), see e.g. [32].

<sup>5</sup> For a PSD matrix  $A \in \mathbb{R}^{n \times n}$ , let its eigen-decomposition be  $A = \sum_{i=1}^n \lambda_i u_i u_i^\top$ , then the Moore-Penrose pseudo-inverse of  $A$  is  $A^+ = \sum_{i: \lambda_i > 0} \frac{1}{\lambda_i} u_i u_i^\top$ .

**Algorithm for Laplacian matrices.** We first present our simpler algorithm for linear systems in Laplacians with a bounded condition number.

► **Theorem 1.1** (Laplacian Solver, see Section 2). *There exists a randomized algorithm, that given input  $\langle G, b, u, \epsilon, \bar{\kappa} \rangle$ , where*

- $G = (V, E)$  is a connected  $d$ -regular graph given as an adjacency list,
- $b \in \mathbb{R}^n$  is in the range of  $L_G$  (equivalently, orthogonal to the all-ones vector),
- $u \in [n]$ ,  $\epsilon > 0$ , and
- $\bar{\kappa} \geq 1$  is an upper bound on the condition number  $\kappa(L_G)$ ,

*the algorithm outputs  $\hat{x}_u \in \mathbb{R}$  with the following guarantee. Letting  $x^* = L_G^+ b$ , we have*

$$\forall u \in [n], \quad \Pr \left[ |\hat{x}_u - x_u^*| \leq \epsilon \cdot \|x^*\|_\infty \right] \geq 1 - \frac{1}{s},$$

*and the algorithm runs in time  $O(d\epsilon^{-2}s^3 \log s)$ , for suitable  $s = \Theta(\bar{\kappa} \log(\epsilon^{-1} \bar{\kappa} n))$ .*

A few extensions of the theorem follow easily from our proof. First, if the algorithm is given also an upper bound  $B_{up}$  on  $\|b\|_0$ , then the expression for  $s$  can be refined by replacing  $n$  with  $B_{up} \leq n$ . Second, we can improve the running time to  $O(\epsilon^{-2}s^3 \log s)$  whenever the representation of  $G$  allows to sample a uniformly random neighbor of a vertex in constant time. Third, the algorithm has an (essentially) cubic dependence on the condition number  $\kappa(L_G)$ , which can be improved to quadratic if we allow a preprocessing of  $G$  (or, equivalently if we only count the number of probes into  $b$ ). Later we show that this quadratic dependence is *near-optimal*.

**Algorithm for SDD matrices.** We further design an algorithm for SDD matrices with bounded condition number. The formal statement, which appears in Theorem 3.1, is a natural generalization of Theorem 1.1 with two differences. One difference is that a natural solution to the system  $Sx = b$  is  $x = S^+ b$ , but our method requires  $S$  to have normalized diagonal entries, and thus we aim at another solution  $x^*$ , constructed as follows. Define

$$D \stackrel{\text{def}}{=} \text{diag}(S_{11}, \dots, S_{nn}) \quad \text{and} \quad \tilde{S} \stackrel{\text{def}}{=} D^{-1/2} S D^{-1/2}, \tag{2}$$

then our linear system can be written as  $\tilde{S}(D^{1/2}x) = D^{-1/2}b$ , which has a solution

$$x^* \stackrel{\text{def}}{=} D^{-1/2} \tilde{S}^+ D^{-1/2} b, \tag{3}$$

which is expressed using the pseudo-inverse of  $\tilde{S}$  rather than of  $S$ .

A second difference is that Theorem 3.1 makes no assumptions about the multiplicity of the eigenvalue 0 of  $\tilde{S}$ , e.g., if  $S$  is a graph Laplacian, then the graph need not be connected. The assumptions needed to achieve a polylogarithmic time, beyond  $\tilde{S}$  having a bounded condition number,<sup>6</sup> are only that a random “neighbor” in the graph corresponding to  $S$  can be sampled quickly, and that  $\frac{\max_{i \in [n]} D_{ii}}{\min_{i \in [n]} D_{ii}} \leq \text{poly}(n)$ , which holds if  $S$  has polynomially-bounded entries.

**Lower Bound for PSD matrices.** Our first lower bound shows that the above guarantees cannot be obtained for a general PSD matrix, even if we are allowed to preprocess the matrix  $S$ , and only count probes into  $b$ . The proof employs a PSD matrix  $S$  that is invertible (i.e., positive definite), in which case the linear system  $Sx = b$  has a unique solution  $x = S^{-1}b$ .

---

<sup>6</sup> We cannot phrase our requirements in terms of  $\kappa(S)$ , because we are not aware of a non-trivial relationship between it and  $\kappa(\tilde{S})$ .

► **Theorem 1.2** (Lower Bound for PSD Systems, see Section 4). *For every large enough  $n$ , there exists an invertible PSD matrix  $S \in \mathbb{R}^{n \times n}$  with uniformly bounded sparsity  $d = O(1)$  and condition number  $\kappa(S) \leq 3$ , and a distinguished index  $u \in [n]$ , which satisfy the following. Every randomized algorithm that, given as input  $b \in \mathbb{R}^n$ , outputs  $\hat{x}_u$  satisfying*

$$\Pr \left[ |\hat{x}_u - x_u^*| \leq \frac{1}{5} \|x^*\|_\infty \right] \geq \frac{6}{7},$$

where  $x^* = S^{-1}b$ , must probe  $n^{\Omega(1/d^2)}$  coordinates of  $b$  (in the worst case).

**Dependence on Condition Number.** The second lower bound shows that our SDD algorithm has a *near-optimal* dependence on the condition number of  $S$ , even if we are allowed to preprocess the matrix  $S$ , and only count probes into  $b$ . The lower bound holds even for Laplacian matrices. Here and throughout, we use  $\tilde{O}(f)$  to hide polylogarithmic factors in  $f$  or in the input size, i.e., it stands for  $O(f \log^{O(1)}(f+n))$ , and similarly for  $\tilde{\Omega}(f)$ .

► **Theorem 1.3** (Lower Bound for Laplacian Systems). *For every large enough  $n$  and  $k \leq O(n^{1/2}/\log n)$ , there exist an unweighted graph  $G = ([n], E)$  with maximum degree 4 and whose Laplacian  $L_G$  has condition number  $\kappa(L_G) = O(k)$ , and a distinguished edge  $(u, v)$  in  $G$ , which satisfy the following. Every randomized algorithm that, given input  $b$  in the range of  $L_G$ , succeeds with probability  $2/3$  to approximate  $x_u - x_v$  within additive error  $\epsilon \|x^*\|$  for  $\epsilon = \Theta(1/\log n)$  and any solution  $x^* \in \mathbb{R}^n$  for  $L_G x = b$ , must probe  $\tilde{\Omega}(k^2)$  coordinates of  $b$  (in the worst case).*

The proof of this result is omitted here but appears in the full version.

**Applications.** An example application of our algorithmic results is computing the effective resistance between a pair of vertices  $u, v$  in a graph  $G$  (given  $u, v$  and  $G$  as input). It is well known that the effective resistance, denoted  $R_{\text{eff}}(u, v)$ , can be expressed as  $x_u - x_v$ , where  $x$  solves  $L_G x = e_u - e_v$ . The spectral-sparsification algorithm of Spielman and Srivastava [33] relies on a near-linear time algorithm (that they devise) for approximating the effective resistances of all edges in  $G$ . For unweighted graphs, there is also a faster algorithm [25] that runs in time  $\tilde{O}(n)$ , which is sublinear in the number of edges, and approximates effective resistances within a larger factor  $\text{polylog}(n)$ . In a  $d$ -regular expander  $G$ , it is the effective resistance of every vertex pair is  $\Theta(1/d)$ , and in this case our algorithm from Theorem 1.1 can quickly compute, for any single pair, an arbitrarily good approximation (factor  $1+\epsilon$ ). Indeed, observe that we can use  $B_{up} = 2$ , hence the running time is  $O(\frac{1}{\epsilon^2} \text{polylog}(\frac{1}{\epsilon}))$ , independently of  $n$ . The additive accuracy is  $\epsilon \|x\|_\infty$ , where  $x \in \mathbb{R}^n$  represents the vertex potentials when imposing a unit of current from  $u$  to  $v$ , or equivalently, imposing a potential difference  $R_{\text{eff}}(u, v)$  between  $u$  and  $v$ , which implies that every  $x_i \in [x_u, x_v]$ . By considering a solution with  $x_u = 0$  we get  $\|x\|_\infty = x_u - x_v = R_{\text{eff}}(u, v)$ , and thus with high probability, the output actually achieves a multiplicative guarantee  $\hat{R}_{\text{eff}}(u, v) \in (1 \pm \epsilon) R_{\text{eff}}(u, v)$ .

## 1.2 Technical Outline

**Algorithms.** Our basic technique relies on a classic idea of von Neumann and Ulam [18, 38] for estimating a matrix inverse by a power series; see Section 1.3 for a discussion of related work. Our starting point is the identity

$$\forall X \in \mathbb{R}^{n \times n}, \|X\| < 1, \quad (I - X)^{-1} = \sum_{t=0}^{\infty} X^t.$$

(Recall that  $\|X\|$  denotes the spectral norm of a matrix  $X$ .) Now given a Laplacian  $L = L_G$  of a  $d$ -regular graph  $G$ , observe that  $\frac{1}{d}L = I - \frac{1}{d}A$ , where  $A$  is the adjacency matrix of  $G$ . Assume for a moment that  $\|\frac{1}{d}A\| < 1$ ; then by the above identity,  $(\frac{1}{d}L)^{-1} = (I - \frac{1}{d}A)^{-1} = \sum_{t=0}^{\infty} (\frac{1}{d}A)^t$ , and the solution of the linear system  $Lx = b$  would be  $x^* = L^{-1}b = \frac{1}{d} \sum_{t=0}^{\infty} (\frac{1}{d}A)^t b$ . The point is that the summands decay exponentially because  $\|(\frac{1}{d}A)^t b\|_2 \leq \|(\frac{1}{d}A)^t\| \cdot \|b\|_2 \leq \|(\frac{1}{d}A)\|^t \cdot \|b\|_2$ . Therefore, we can estimate  $x_u^*$  using the first  $t_0$  terms, i.e.,  $\hat{x}_u = e_u^\top \frac{1}{d} \sum_{t=0}^{t_0} (\frac{1}{d}A)^t b$ , where  $t_0$  is logarithmic (with base  $\|\frac{1}{d}A\|^{-1} > 1$ ). In order to compute each term  $e_u^\top \frac{1}{d} (\frac{1}{d}A)^t b$ , observe that  $e_u^\top (\frac{1}{d}A)^t e_w$  is exactly the probability that a random walk of length  $t$  starting at  $u$  will end at vertex  $w$ . Thus, if we perform a random walk of length  $t$  starting at  $u$ , and let  $z$  be its (random) end vertex, then

$$\mathbb{E}[b_z] = \sum_{w \in V} e_u^\top (\frac{1}{d}A)^t e_w b_w = e_u^\top (\frac{1}{d}A)^t b.$$

If we perform several random walks (specifically,  $\text{poly}(t_0, \frac{1}{\epsilon})$  walk suffice), average the resulting  $b_z$ 's, and then multiply by  $\frac{1}{d}$ , then with high probability, we will obtain a good approximation to  $e_u^\top \frac{1}{d} (\frac{1}{d}A)^t b$ .

As a matter of fact, we have a non-strict inequality  $\|\frac{1}{d}A\| \leq 1$ , because of the all-ones vector  $\vec{1} \in \mathbb{R}^n$ . Nevertheless, we can still get a meaningful result if all eigenvalues of  $A$  except for the largest one are smaller than  $d$  (equivalently, the graph  $G$  is connected). First, we get rid of any negative eigenvalues by the standard trick of considering  $(dI + A)/2$  instead of  $A$ , which is equivalent to adding  $d$  self-loops at every vertex. Second, we may assume  $b$  is orthogonal to  $\vec{1}$  (otherwise the linear system has no solution), and while the linear system  $Lx = b$  has infinitely many solutions, we estimate the specific solution  $x^* \stackrel{\text{def}}{=} L^+ b$  (recall  $L$  is PSD) by  $\frac{1}{d} \sum_{t=0}^{t_0} (\frac{1}{d}A)^t b$ . Indeed, the idealized analysis above still applies by restricting all our calculations to the subspace orthogonal to  $\vec{1}$ . This is carried out in Theorem 1.1.

To generalize the above approach to SDD matrices, we face three issues. First, due to the irregularity of general SDD matrices, it is harder to properly define the equivalent random walk matrix. We resolve this by normalizing the SDD matrix  $S$  into  $\tilde{S}$  defined in (2), and solving the equivalent (normalized) system  $\tilde{S}(D^{1/2}x) = D^{-1/2}b$ . Second, general SDD matrices can have *positive* off-diagonal elements, in contrast to Laplacians. To address this, we interpret such entries as negative-weight edges, and employ random walks that “remember” the signs of the traversed edges. Third, diagonal elements may strictly dominate their row, which we address by terminating the random walk early with some positive probability.

**Lower Bound: Polynomial Time for PSD Matrices.** We first discuss our lower bound for PSD matrices, which is one of the main contributions of our work. It exhibits a family of matrices  $S$  for which estimating a coordinate  $x_u^*$  of the solution  $x^* = S^{-1}b$  requires  $n^{\Omega(1)}$  probes into the input  $b$ .

Without the sparsity constraint on  $S$ , one can deduce such a lower bound via a reduction from the communication complexity of the *Vector in Subspace Problem* (VSP), in which Alice has an  $n/2$ -dimensional subspace  $H \subset \mathbb{R}^n$ , Bob has a vector  $b \in \mathbb{R}^n$ , and their goal is to determine whether  $b \in H$  or  $b \in H^\perp$ . The randomized communication complexity of this promise problem is between  $\Omega(n^{1/3})$  [23] and  $O(\sqrt{n})$  [28] (while for quantum communication it is  $O(\log n)$ ). To reduce this problem to linear-system solvers, let  $P_H \in \mathbb{R}^{n \times n}$  be the projection operator onto the subspace  $H$ , and set  $S = I + P_H$ . Consider the system  $Sx = b$ , and notice that Alice knows  $S$  and Bob knows  $b$ . It is easy to see that the unique solution  $x^*$  is either  $b$  or  $\frac{1}{2}b$ , depending on whether  $b \in H^\perp$  or  $b \in H$ . Alice and Bob could use a solver that makes few probes to  $b$ , as follows. Bob would pick an index  $u \in [n]$  that maximizes



$|b_u|$  (and thus also  $|x_u|$ ), and send it to Alice. She would then apply the solver, asking Bob for only a few entries of  $b$ , to estimate  $x_u$  within additive error  $\frac{1}{2}\|x\|_\infty$ , which suffices to distinguish the two cases. This matrix  $S$  is PSD with condition number  $\kappa(S) \leq 2$ . However, it is dense.

We thus revert to a different approach of proving it from basic principles. Our high-level idea is to take a  $2d$ -regular expander and assign to its edges random signs ( $\pm 1$ ) that are balanced everywhere, namely, at every vertex the incident edges are split evenly between positive and negative. The signed adjacency matrix  $A \in \{-1, 0, +1\}^{n \times n}$  should have spectral norm  $\mu \stackrel{\text{def}}{=} \|A\| = O(\sqrt{d})$ , and then instead of the (signed) Laplacian  $L = (2d)I - A$ , we consider  $S = 2\mu I - A$ , which is PSD with condition number  $\kappa(S) \leq 3$ , as well as invertible and sparse. Now following arguments similar to our algorithm, we can write  $S^{-1}$  as a power series of the matrix  $A$ , and express coordinate  $x_u^*$  of the solution  $x^* = S^{-1}b$  via  $\mathbb{E}_z[b_z]$  where  $z$  is the (random) end vertex of a random walk that starts at  $u$  and its length is bounded by some  $t_0$  (performed in the “signed” graph corresponding to  $A$ ). Now if the graph around  $u$  looks like a tree (e.g., it has high girth), then not-too-long walks are highly symmetric and easy to count. We now let  $b_v$  be non-zero only at vertices  $v$  at distance exactly  $t_0$  from  $u$ , and for these vertices set  $b_v \in \{+1, -1\}$  at random but with a small bias  $\delta$  towards one of the values. Some calculations show that  $\text{sgn}(\mathbb{E}_z[b_z])$ , and consequently  $\text{sgn}(x_u^*)$ , will be according to our bias (with high probability), however discovering this  $\text{sgn}(x_u^*)$  via probes to  $b$  is essentially the problem of learning a biased coin, which requires  $\Omega(\delta^{-2})$  coin observations. An additional technical obstacle is to bound  $\|x^*\|_\infty$ , so that we can argue that an  $\frac{1}{5}\|x^*\|_\infty$ -additive error to  $x_u^*$  will not change its sign. Overall, we show we can set  $t_0 = \Omega(\log_d n)$  and  $\delta \approx ((2d - 1)^{t_0})^{-1/2}$ , thus concluding that the algorithm must observe  $\Omega(\delta^{-2}) = n^{\Omega(1)}$  entries of  $b$ .

It is instructive to ask where in the above argument is it crucial to have  $\mu = O(\sqrt{d})$ , because if it were valid also for  $\mu = d$ , then it would hold also for the SDD matrix  $S = 2\mu I - A$ , and contradict our own algorithm for SDD matrices. The answer is that  $\mu \ll 2d$  is required to bound  $\|x^*\|_\infty$  in Lemma 4.8.

**Lower Bound: Quadratic Dependence on Condition Number.** We now outline the ideas to prove the  $\tilde{\Omega}(\kappa^2)$  lower bound even for Laplacian systems with condition number  $\kappa$ . First, it is relatively straightforward to prove that a *linear* dependence on the condition number is necessary. Indeed, consider a dumbbell graph, namely, two 3-regular expanders connected by a bridge edge  $(u, v)$ , and suppose one need to estimate  $x_u^* - x_v^*$ . For input  $b = e_i - e_j$ , the value of  $x_u^* - x_v^*$  is non-zero iff vertices  $i, j$  are on opposite sides of the bridge, and determining the latter requires  $\Omega(n)$  probes into  $b$ . Since this graph has condition number  $O(n)$ , we obtain an  $\Omega(\kappa)$  lower bound.

The quadratic lower bound requires both a different graph and a different vector  $b$ . We use the following graph  $G$  with condition number  $O(k)$ : take two 3-regular expanders and connect them with  $n/k$  “bridge edges”. The vector  $b \in \{-1, +1\}^n$  is *dense* and in particular it is either: 1) balanced, i.e.,  $\sum b_i$  on each expander is zero; or 2) unbalanced, i.e., each  $b_i \in \{+1, -1\}$  at random with a bias  $p \approx 1/k$  towards  $+1$  on the first expander, and towards  $-1$  on the second one. Now, as above, it is simple to prove that: 1) in the balanced case, the average of  $x_u^* - x_v^*$  over all bridge edges  $(u, v)$  must be zero; and 2) in the unbalanced case, the same average must be  $\Omega(1)$ . The main challenge is that the actual values might differ from the average – e.g., even in the balanced case, each bridge edge  $(u, v)$  will likely have non-zero value of  $x_u^* - x_v^*$ . Nonetheless, we manage to prove an upper bound on the maximum value of  $|x_u^* - x_v^*|$  over all edges  $(u, v)$  (as in the previous lower bound, we need to bound  $\|x^*\|_\infty$  as well). For the latter, we need to again analyze  $\mathbb{E}_z[b_z]$  where  $z$  is the end vertex of a random walk of some fixed length  $i \geq 1$  starting from  $u$  in the graph  $G$ . Since

the vector  $b$  is not symmetric over the graph  $G$ , a direct analysis seems hard – instead we estimate  $\mathbb{E}_z[b_z]$  via a coupling of such walks in  $G$  with random walks in an expander, which is amenable to a direct analysis.

### 1.3 Related Work

The idea of approximating the inverse  $(I - X)^{-1} = \sum_{t=0}^{\infty} X^t$  (for  $\|X\| < 1$ ) by random walks dates back to von Neumann and Ulam [18, 38]. While we approximate each power  $X^t$  by separate random walks of length  $t$  and truncate the tail (powers above some  $t_0$ ), their method employs random walks whose length is random and whose expectation gives exactly the infinite sum, achieved by assigning some probability to terminate the walk at each step, and weighting the contributions of the walks accordingly (to correct the expectation).

The idea of approximating a generalized inverse  $L^*$  of  $L = dI - A$  by the truncated series  $\frac{1}{d} \sum_{t=0}^{t_0} (\frac{1}{d}A)^t$  on directions that are orthogonal to the all-ones vector was recently used by Doron, Le Gall, and Ta-Shma [16] to show that  $L^*$  can be approximated in probabilistic log-space. However, since they wanted to output  $L^*$  explicitly, they could not ignore the all-ones direction and they needed to relate  $L^*$  to  $\frac{1}{d} \sum_{t=0}^{\infty} (\frac{1}{d}A)^t$  by “peeling off” the all-ones direction, inverting using the infinite sum formula, and then adding back the all-ones direction.

The idea of estimating powers of a normalized adjacency matrix  $\frac{1}{d}A$  (or more generally, a stochastic matrix) by performing random walks is well known, and was used also in [16] mentioned above, and in [17]. Chung and Simpson [12] used it in a context that is related to ours, of solving a Laplacian system  $L_G x = b$ , but with a boundary condition, namely, a constraint that  $x_i = b_i$  for all  $i$  in the support of  $b$ . Their algorithm solves for a subset of the coordinates  $W \subseteq V$ , i.e., it approximates  $x|_W$  (the restriction of  $x$  to coordinates in  $W$ ) where  $x$  solves  $Lx = b$  under the boundary condition. They relate the solution  $x$  to the Dirichlet heat-kernel PageRank vector, which in turn is related to an infinite power series of a transition matrix (specifically, to  $f^\top e^{-t(I-P_W)} = e^{-t} f^\top \sum_{k=0}^{\infty} \frac{t^k}{k!} P_W^k$  where  $P_W$  is the transition matrix of the graph induced by  $W$ ,  $t \in \mathbb{R}$ , and  $f \in \mathbb{R}^{|W|}$ ), and their algorithm uses random walks to approximate the not-too-large powers of the transition matrix, proving that the remainder of the infinite sum is small enough.

Recently, Shyamkumar, Banerjee and Lofgren [30] considered a related matrix-power problem, where the input is a matrix  $A \in \mathbb{R}^{n \times n}$ , a power  $\ell \in \mathbb{N}$ , a vector  $z \in \mathbb{R}^n$ , and an index  $u \in [n]$ , and the goal is to compute coordinate  $u$  of  $A^\ell z$ . They devised for this problem a sublinear (in  $\text{nnz}(A)$ ) algorithm, under some bounded-norm conditions and assuming  $u \in [n]$  is uniformly random. Their algorithm relies, in part, on von Neumann and Ulam’s technique of computing matrix powers using random walks, but of prescribed length. It can be shown that approximately solving positive definite systems for a particular coordinate is reducible to the matrix-power problem.<sup>7</sup> However, in contrast to our results, their expected running time is polynomial in the input size, namely  $\text{nnz}(A)^{2/3}$ , and holds only for a random  $u \in [n]$ .

**Comparison with PageRank.** An example application of our results is computing quickly the PageRank (defined in [8]) of a single node in an undirected  $d$ -regular graph. Recall that the PageRank vector of an  $n$ -vertex graph with associated transition matrix  $P$  is the solution

<sup>7</sup> Let  $Ax = b$  be a linear system where  $A$  is positive definite. Let  $\lambda$  be the largest eigenvalue of  $A$ . Let  $A' \stackrel{\text{def}}{=} \frac{1}{2\lambda}A$  and  $b' \stackrel{\text{def}}{=} \frac{1}{2\lambda}b$ . Consider the equivalent system  $(I - (I - A'))x = b'$ . As the eigenvalues of  $A'$  are in  $(0, 1/2]$ , the eigenvalues of  $I - A'$  are in  $[1/2, 1)$ . Thus, the solution to the linear system is given by  $x = (I - (I - A'))^{-1}b' = \sum_{t=0}^{\infty} (I - A')^t b'$ . Therefore, we can approximate  $x_u$  by truncating the infinite sum at some  $t_0$  and approximating each power  $t < t_0$  by the algorithm for the matrix-power problem.

to the linear system  $x = \frac{1-\alpha}{n}\vec{1} + \alpha Px$ , where  $0 < \alpha < 1$  is a given parameter. In personalized PageRank, one replaces  $\frac{1}{n}\vec{1}$  (the uniform distribution) with some  $b \in \mathbb{R}^n$ , e.g., a standard basis vector. Equivalently,  $x$  solves the system  $Sx = \frac{1-\alpha}{n}\vec{1}$  where  $S = I - \alpha P$  is an SDD matrix with 1's on the diagonal. As all eigenvalues of  $P$  are of magnitude at most 1 (recall  $P$  is a transition matrix), all eigenvalues of  $I - \tilde{S} = I - S = \alpha P$  are of magnitude at most  $\alpha$ , and the running time guaranteed by Theorem 3.1 is logarithmic (with base  $\frac{2}{\alpha+1}$ ).

Algorithms for the PageRank model were studied extensively, and usually consider arbitrary (and even directed) graphs. In particular, the sublinear algorithms of [7] approximate the PageRank of a vertex using  $\tilde{O}(n^{2/3})$  queries, or using  $\tilde{O}((n\Delta)^{1/2})$  queries when the maximum degree is  $\Delta$ . Another example is the heavy-hitters algorithm of [6], which reports all vertices whose approximate PageRank exceeds a threshold  $T$  in sublinear time  $\tilde{O}(1/\Delta)$ , when PageRanks are viewed as probabilities and sum to 1. Other work explores connections to other graph problems, including for instance using PageRank algorithms to approximate effective resistances [13], the PageRank vector itself, and computing sparse cuts [4].

**Local Algorithms.** Our algorithms in Theorems 1.1 and 3.1 are *local* in the sense that they query a small portion of their input, usually around the input vertex, when viewed as graph algorithms. Local algorithms for graph problems were studied in several contexts, like graph partitioning [31, 5], Web analysis [10, 3], and distributed computing [34]. Rubinfeld, Tamir, Vardi, and Xie [29] introduced a formal concept of *Local Computation Algorithms* that requires consistency between the local outputs of multiple executions (namely, these local outputs must all agree with a single global solution). As explained earlier, our problem formulation (1) follows this consistency requirement.

## 1.4 Future Work

One may study alternative ways of defining the problem of solving a linear system in sublinear time, in particular if the algorithm can access  $b$  in a different way. For example, similarly to assumptions and guarantees in [35], the goal may be to produce an  $\ell_2$ -sample from the solution  $x$  (i.e., report a random index in  $[n]$  such that the probability of each coordinate  $i \in [n]$  is proportional to  $x_i^2$ ) assuming oracle access to an  $\ell_2$ -sampler from  $b \in \mathbb{R}^n$ , i.e., use an  $\ell_2$ -sampler for  $b$  to construct an  $\ell_2$ -sampler for  $x$ . Another version of the problem may ask to produce heavy hitters in  $x$ , assuming, say,<sup>8</sup> heavy hitters in  $b$  (which may be useful for the PageRank application). We leave these extensions as interesting open questions, focusing here on the classical access mode to  $b$ , via queries to its coordinates.

Another variation one may consider is to bound the error using a norm other than  $\ell_\infty$ , like  $\|y\|_S \stackrel{\text{def}}{=} \sqrt{y^\top S y}$  used in [31]. For example, if  $S$  is the Laplacian of a  $d$ -regular expander and  $y$  is orthogonal to the all-ones vector, then  $\|y\|_S = \Theta(\sqrt{d}\|y\|_2)$ , which might exceed  $\|y\|_\infty$  significantly even for constant  $d$ . Nevertheless, our requirement  $\|\hat{x} - x^*\|_\infty \leq \epsilon\|x^*\|_\infty$  is generally incomparable to  $\|\hat{x} - x^*\|_2 \leq \epsilon\|x^*\|_2$ .

## 2 Laplacian Solver (for Regular Graphs)

In this section we shall prove Theorem 1.1. The ensuing description deals mostly with a slightly simplified scenario, where the algorithm is given not one but two vertices  $u, v \in [n]$ , and returns an approximation  $\hat{\delta}_{u,v}$  to  $x_u - x_v$  with a slightly different error bound, see

<sup>8</sup> This kind of oracle seems necessary even when  $S = I$ .

---

**Algorithm 1** Solve-Linear-Laplacian.

---

**input** :  $d$ -regular graph  $G$ ; vector  $b$ ;  $\|b\|_0$ ; vertices  $u, v$ ; accuracy parameter  $\epsilon$ ; and  $\mu_2$

**output**: estimate  $\hat{\delta}_{u,v}$  for  $x_u - x_v$

- 1 set  $s = \frac{\log(2\sqrt{2}\epsilon^{-1} \frac{d}{\mu_2} \sqrt{\|b\|_0})}{\log(\frac{d}{d-\mu_2})}$  and  $\ell = O((\frac{\epsilon}{4s})^{-2} \log s)$
  - 2 **for**  $t = 0, 1, \dots, s - 1$  **do**
  - 3     Perform  $\ell$  independent random walks of length  $t$  starting at  $u$ , and let  $u_1^{(t)}, \dots, u_\ell^{(t)}$  be their end vertices. Independently, perform  $\ell$  independent random walks of length  $t$  starting at  $v$ , and let  $v_1^{(t)}, \dots, v_\ell^{(t)}$  be their end vertices.
  - 4     set  $\hat{\delta}_{u,v}^{(t)} = \frac{1}{\ell} \sum_{i \in [\ell]} (b_{u_i^{(t)}} - b_{v_i^{(t)}})$
  - 5 **return**  $\hat{\delta}_{u,v} = \frac{1}{d} \sum_{t=0}^{s-1} \hat{\delta}_{u,v}^{(t)}$
- 

Theorem 2.5 for the precise statement. The advantage is that if  $G$  is connected, all solutions  $x$  give rise to a unique value for  $x_u - x_v$ . We will then explain the modifications required to prove Theorem 1.1 (which actually follows also from our more general Theorem 3.1).

Let  $G = (V = [n], E)$  be a connected  $d$ -regular graph with adjacency matrix  $A \in \mathbb{R}^{n \times n}$ . Let the eigenvalues of  $A$  be  $d = \lambda_1 > \lambda_2 \geq \dots \geq \lambda_n$ , and let their associated orthonormal eigenvectors be  $u_1, \dots, u_n$ . Then  $u_1 = \frac{1}{\sqrt{n}} \cdot \vec{1} \in \mathbb{R}^n$ , and we can write  $A = U\Lambda U^\top$  where  $U = [u_1 \ u_2 \ \dots \ u_n]$  is unitary and  $\Lambda = \text{diag}(\lambda_1, \dots, \lambda_n)$ . For  $u, v \in [n]$ , let  $\chi_{u,v} \stackrel{\text{def}}{=} e_u - e_v$  where  $e_i$  is the  $i$ -th standard basis vector. Then the Laplacian of  $G$  is given by

$$L \stackrel{\text{def}}{=} \sum_{uv \in E} \chi_{u,v} \chi_{u,v}^\top = dI - A = U(dI - \Lambda)U^\top.$$

Observe that  $L$  does not depend on the orientation of each edge  $uv$ , and that  $\mu_2 \stackrel{\text{def}}{=} d - \lambda_2$  is the smallest non-zero eigenvalue of  $L$ . The Moore-Penrose pseudo-inverse of  $L$  is

$$L^+ \stackrel{\text{def}}{=} U \cdot \text{diag}(0, (d - \lambda_2)^{-1}, \dots, (d - \lambda_n)^{-1}) \cdot U^\top.$$

We assume henceforth that all eigenvalues of  $A$  are non-negative. At the end of the proof, we will remove this assumption (by adding self-loops).

The idea behind the next fact is that  $L = d(I - \frac{1}{d}A)$ , and  $\frac{1}{d}A$  has norm strictly smaller than one when operating on the subspace that is orthogonal to the all-ones vector, and hence, the formula  $(I - X)^{-1} = \sum_{t=0}^{\infty} X^t$  for  $\|X\| < 1$  is applicable for the span of  $\{u_2, \dots, u_n\}$ .

► **Fact 2.1.** For every  $x \in \mathbb{R}^n$  that is orthogonal to the all-ones vector,  $L^+x = \frac{1}{d} \sum_{t=0}^{\infty} (\frac{1}{d}A)^t x$ .

**Proof.** It suffices to prove the claim for each of  $u_2, \dots, u_n$  as the fact will then follow by linearity. Fix  $i \in \{2, \dots, n\}$ . Then since  $|\frac{\lambda_i}{d}| < 1$ ,

$$\sum_{t=0}^{\infty} \left(\frac{1}{d}A\right)^t u_i = \sum_{t=0}^{\infty} \left(\frac{\lambda_i}{d}\right)^t u_i = \frac{1}{1 - \frac{\lambda_i}{d}} u_i = \frac{d}{d - \lambda_i} u_i = dL^+ u_i. \quad \blacktriangleleft$$

We now describe an algorithm that on input  $b \in \mathbb{R}^n$  that is orthogonal to the all-ones vector, and two vertices  $u \neq v \in [n]$ , returns an approximation  $\hat{\delta}_{u,v}$  to  $x_u - x_v$ , where  $x$  solves  $Lx = b$ . As  $G$  is connected, the null space of  $L$  is equal to  $\text{span}\{\vec{1}\}$  and hence  $x_u - x_v$  is uniquely defined, and can be written as  $x_u - x_v = \chi_{u,v}^\top L^+ b$ .

► **Claim 2.2.** For  $b$  that is orthogonal to the all-ones vector and  $s = \frac{\log(2\sqrt{2}\epsilon^{-1} \frac{d}{\mu_2} \sqrt{\|b\|_0})}{\log(\frac{d}{d-\mu_2})}$ ,  
 $|\chi_{u,v}^\top L^+ b - \chi_{u,v}^\top \frac{1}{d} \sum_{t=0}^{s-1} (\frac{1}{d} A)^t b| \leq \frac{\epsilon}{2d} \|b\|_\infty$ .

**Proof.** Using Fact 2.1,

$$\chi_{u,v}^\top L^+ b - \chi_{u,v}^\top \frac{1}{d} \sum_{t=0}^{s-1} (\frac{1}{d} A)^t b = \chi_{u,v}^\top \frac{1}{d} \sum_{t=s}^{\infty} (\frac{1}{d} A)^t b,$$

and thus

$$|\chi_{u,v}^\top L^+ b - \chi_{u,v}^\top \frac{1}{d} \sum_{t=0}^{s-1} (\frac{1}{d} A)^t b| \leq \|\chi_{u,v}^\top\|_2 \cdot \left\| \frac{1}{d} \sum_{t=s}^{\infty} (\frac{1}{d} A)^t b \right\|_2.$$

We know that  $\|\chi_{u,v}^\top\|_2 = \sqrt{2}$ , so it remains to bound  $\left\| \frac{1}{d} \sum_{t=s}^{\infty} (\frac{1}{d} A)^t b \right\|_2$ . Decomposing  $b = \sum_{i=2}^n c_i u_i$  we get that  $\sum_{i=2}^n c_i^2 = \|b\|_2^2$  and

$$\sum_{t=s}^{\infty} (\frac{1}{d} A)^t b = \sum_{i=2}^n c_i u_i \sum_{t=s}^{\infty} \left(\frac{\lambda_i}{d}\right)^t = \sum_{i=2}^n \frac{(\frac{\lambda_i}{d})^s}{1 - \frac{\lambda_i}{d}} c_i u_i = d \sum_{i=2}^n \frac{(\frac{\lambda_i}{d})^s}{d - \lambda_i} c_i u_i.$$

Hence,

$$\left\| \frac{1}{d} \sum_{t=s}^{\infty} (\frac{1}{d} A)^t b \right\|_2^2 = \sum_{i=2}^n \left( \frac{(\frac{\lambda_i}{d})^s}{d - \lambda_i} \right)^2 c_i^2 \|u_i\|_2^2 \leq \left( \frac{(\frac{\lambda_2}{d})^s}{d - \lambda_2} \right)^2 \sum_{i=2}^n c_i^2 = \left( \frac{(1 - \frac{\mu_2}{d})^s}{\mu_2} \right)^2 \|b\|_2^2,$$

where the first equality is because the  $u_i$ 's are orthogonal. Altogether,

$$|\chi_{u,v}^\top L^+ b - \chi_{u,v}^\top \frac{1}{d} \sum_{t=0}^{s-1} (\frac{1}{d} A)^t b| \leq \sqrt{2} \frac{(1 - \frac{\mu_2}{d})^s}{\mu_2} \|b\|_2 \leq \sqrt{2} \frac{(1 - \frac{\mu_2}{d})^s}{\mu_2} \sqrt{\|b\|_0} \cdot \|b\|_\infty = \frac{\epsilon}{2d} \|b\|_\infty,$$

as claimed. ◀

► **Claim 2.3.**  $\Pr \left[ |\hat{\delta}_{u,v} - \chi_{u,v}^\top \frac{1}{d} \sum_{t=0}^{s-1} (\frac{1}{d} A)^t b| > \frac{\epsilon}{2d} \|b\|_\infty \right] \leq \frac{1}{s}$ .

**Proof.** Observe that  $e_u^\top (\frac{1}{d} A)^t$  is a probability vector over  $V$ , and  $e_u^\top (\frac{1}{d} A)^t e_w$  is exactly the probability that a random walk of length  $t$  starting at  $u$  will end at  $w$ . Thus, for every  $t \in \{0, 1, \dots, s-1\}$  and  $i \in [l]$ , we have

$$\mathbb{E}[b_{u_i^{(t)}}] = \sum_{w \in [n]} e_u^\top (\frac{1}{d} A)^t e_w b_w = e_u^\top (\frac{1}{d} A)^t b,$$

and similarly  $\mathbb{E}[b_{v_i^{(t)}}] = e_v^\top (\frac{1}{d} A)^t b$ . By a union bound over Hoeffding bounds, with probability at least  $1 - \frac{1}{s}$ , for every  $t \in \{0, 1, \dots, s-1\}$ , we have  $|\frac{1}{\ell} \sum_{i \in [l]} b_{u_i^{(t)}} - e_u^\top (\frac{1}{d} A)^t b| \leq \frac{\epsilon}{4s} \|b\|_\infty$  and  $|\frac{1}{\ell} \sum_{i \in [l]} b_{v_i^{(t)}} - e_v^\top (\frac{1}{d} A)^t b| \leq \frac{\epsilon}{4s} \|b\|_\infty$ . Recalling that  $\hat{\delta}_{u,v} = \frac{1}{d} \sum_{t=0}^{s-1} \frac{1}{\ell} \sum_{i \in [l]} (b_{u_i^{(t)}} - b_{v_i^{(t)}})$ , with probability at least  $1 - \frac{1}{s}$  we have  $|\hat{\delta}_{u,v} - \chi_{u,v}^\top \frac{1}{d} \sum_{t=0}^{s-1} (\frac{1}{d} A)^t b| \leq \frac{\epsilon}{2d} \|b\|_\infty$ , as claimed. ◀

Combining Claim 2.2 and Claim 2.3 we get that (with probability  $1 - \frac{1}{s}$ )  $|\hat{\delta}_{u,v} - \chi_{u,v}^\top L^+ b| \leq \frac{\epsilon}{d} \|b\|_\infty$ . Now, as  $x$  solves  $Lx = b$ , for every  $i \in [n]$  we have  $\sum_{j \in N(i)} (x_i - x_j) = b_i$  where  $N(i)$  is the set of neighbors  $\{j : ij \in E\}$ , which implies that for some neighbor  $j$  of  $i$ , it holds that  $|x_i - x_j| \geq \frac{|b_i|}{d}$ . Therefore,  $\max_{ij \in E} |x_i - x_j| \geq \frac{1}{d} \|b\|_\infty$ . We conclude that  $|\hat{\delta}_{u,v} - \chi_{u,v}^\top L^+ b| \leq \epsilon \cdot \max_{ij \in E} |x_i - x_j|$ . We now turn to the running time of Algorithm 1,

### 3:12 On Solving Linear Systems in Sublinear Time

which is dominated by the time it takes to perform the random walks. There are  $2s \cdot \ell$  random walks in total. The random walks do not need to be independent for different values of  $t$  (as we applied a union bound over the different  $t$ ), we can extend, at each iteration  $t$ , the  $2\ell$  respective random walks constructed at iteration  $t-1$  by an extra step in time  $O(d)$  (recall we assume  $G$  is given as an adjacency list), obtaining a total runtime  $O(s \cdot \ell \cdot d) = O(d\epsilon^{-2}s^3 \log s)$ . To simplify the expression for  $s$ , we need the following bound.

► **Fact 2.4.** For all  $\delta \in (0, 1)$ ,  $\frac{1}{\ln(1-\delta)^{-1}} \leq \frac{1}{\delta}$ .

**Proof.** We need to show that  $\delta \leq \ln(1-\delta)^{-1}$ , or equivalently,  $e^{-\delta} \geq 1-\delta$ , which is well known. ◀

Applying Fact 2.4 to  $\delta = \frac{\mu_2}{d}$ , we have  $s \leq \frac{d}{\mu_2} \log(2\sqrt{2}\epsilon^{-1} \frac{d}{\mu_2} \sqrt{\|b\|_0})$ , and conclude the following.

► **Theorem 2.5.** Given an adjacency list of a connected  $d$ -regular  $n$ -vertex graph  $G$ , a vector  $b \in \mathbb{R}^n$  that is orthogonal to the all-ones vector, vertices  $u, v \in [n]$ , and scalars  $\|b\|_0$ ,  $\epsilon > 0$ , and  $\mu_2 = d - \lambda_2 > 0$ , Algorithm 1 outputs  $\hat{\delta}_{u,v} \in \mathbb{R}$  satisfying

$$\Pr \left[ \left| \hat{\delta}_{u,v} - \chi_{u,v}^\top L^+ b \right| \leq \epsilon \cdot \max_{ij \in E} |x_i - x_j| \right] \geq 1 - \frac{1}{s},$$

in time  $O(d\epsilon^{-2}s^3 \log s)$  for  $s = O(\frac{d}{\mu_2} \log(\epsilon^{-1} \frac{d}{\mu_2} \|b\|_0))$ .

► **Remark.** If we allow preprocessing of  $G$ , the runtime of Algorithm 1 can be reduced to  $O(\epsilon^{-2}s^2)$ , as follows. At the preprocessing phase, compute  $(\frac{1}{d}A)^t$  for all powers  $t \leq s$ . Then, instead of approximating  $e_u^\top (\frac{1}{d}A)^t b$  for all powers  $t \leq s$ , sample a uniform  $t \in \{0, 1, \dots, s\}$ , and then, in  $O(1)$  time (because the probability vector is precomputed, see [37]), sample  $z \in [n]$  based on the probability vector  $e_u^\top (\frac{1}{d}A)^t$ , and finally, output  $\frac{s+1}{d} b_z$ . The expectation of the output is  $\frac{1}{d} \sum_{t=0}^s (\frac{1}{d}A)^t b$ . As for concentration, since the output is in  $[-\frac{s+1}{d} \cdot \|b\|_\infty, \frac{s+1}{d} \cdot \|b\|_\infty]$ , by the Hoeffding bound,  $O(\epsilon^{-2}s^2)$  many repetitions suffice to obtain (with constant probability) an approximation with additive error  $\frac{\epsilon}{2d} \|b\|_\infty$  (as in Claim 2.3).

We still need to show how to remove the assumption that  $A$  has no negative eigenvalues. Given an adjacency matrix  $A$  which might have negative eigenvalues, consider the PSD matrix  $A' = A + dI$ , which is the adjacency matrix of the  $2d$ -regular graph  $G'$  obtained from  $G$  by adding  $d$  self-loops to each vertex. Observe that  $A' = U(\Lambda + dI)U^\top$  and we can write  $L = dI - A = (2dI - A')$ , and thus, similarly to Fact 2.1,  $L^+ x = \frac{1}{2d} \sum_{t=0}^{\infty} (\frac{1}{2d}A')^t x$ , for  $x \in \mathbb{R}^n$  that is orthogonal to the all-ones vector. Therefore, if we use  $A'$  (which is PSD) to guide Algorithm 1's random walks (i.e., at each step of a walk, with probability 1/2 the walk stays put and with probability 1/2 it moves to a uniform neighbor in  $G$ ) and apply Claims 2.2 and 2.3 (which apply even when  $A$  has self-loops), an estimate  $\hat{\delta}_{u,v}$  satisfying with high probability  $|\frac{1}{2}\hat{\delta}_{u,v} - \chi_{u,v}^\top L^+ b| \leq \epsilon \max_{ij \in E} |x_i - x_j|$  is obtained. When running Algorithm 1 on  $G'$ , the term  $s$  evaluates to  $O(\frac{2d}{2d - (\lambda_2 + d)} \log(\epsilon^{-1} \frac{2d}{2d - (\lambda_2 + d)} \|b\|_0)) = O(\frac{d}{\mu_2} \log(\epsilon^{-1} \frac{d}{\mu_2} \|b\|_0))$ , thus, leaving the guarantee of Theorem 2.5 intact (up to constant factors).

**Proof of Theorem 1.1.** The theorem follows by a simple modifications to the analysis above. Observe that the analysis in Claims 2.2 and 2.3 holds also when replacing  $\mu_2$  by a lower bound on  $\mu_2$ , which in turn is easy to derive from the upper bound  $\bar{\kappa}$  given in the input and  $d$  given as part of input  $G$ . Similarly,  $\|b\|_0$  can be replaced by an upper bound  $B_{up} \geq \|b\|_0$ .

To handle one vertex  $u \in [n]$  instead of two vertices  $u, v \in [n]$ , ignore the part dealing with  $v$  in Algorithm 1, and modify the analysis in the two aforementioned claims to use  $e_u$  instead of  $\chi_{u,v}$ . The error bound obtained from combining these lemmas is  $\frac{\epsilon}{d} \|b\|_\infty$ , but since each  $|b_i| = |\sum_j L_{ij} x_j| \leq \sum_j |L_{ij}| \cdot \|x\|_\infty = 2d \|x\|_\infty$ , we can bound the error by  $\frac{\epsilon}{d} \|b\|_\infty \leq 2\epsilon \|x\|_\infty$ . ◀



### 3 An SDD Solver

In this section we prove the following theorem for solving linear systems in SDD matrices. To generalize from Laplacians of regular graphs to SDD matrices, we face several issues as described in Section 1.2. We use the notation defined in (2)-(3).

► **Theorem 3.1** (SDD Solver). *There exists a randomized algorithm, that given input  $\langle S, b, u, \epsilon, \bar{\lambda}_{up} \rangle$ , where*

- $S \in \mathbb{R}^{n \times n}$  is an SDD matrix,
- $b \in \mathbb{R}^n$  is in the range of  $S$  (equivalently, orthogonal to the kernel of  $S$ ),
- $u \in [n]$ ,  $\epsilon > 0$ , and
- $\bar{\kappa} \geq 1$  is an upper bound on the condition number  $\kappa(\tilde{S})$ ,

*this algorithm outputs  $\hat{x}_u \in \mathbb{R}$  with the following guarantee. Suppose  $x^*$  is the solution for  $Sx = b$  given in (3), then*

$$\forall u \in [n], \quad \Pr \left[ |\hat{x}_u - x_u^*| \leq \epsilon \|x^*\|_\infty \right] \geq 1 - \frac{1}{s}$$

*for suitable  $s = O(\bar{\kappa} \log(\epsilon^{-1} \bar{\kappa} \|b\|_0 \cdot \frac{\max_{i \in [n]} D_{ii}}{\min_{i \in [n]} D_{ii}}))$ . The algorithm runs in time  $O(f \epsilon^{-2} s^3 \log s)$ , where  $f$  is the time to make a step in a random walk in the weighted graph formed by the non-zeros of  $S$ .*

Due to space constraints, the proof is omitted from this version.

### 4 Lower Bound for PSD Matrices

In this section we prove Theorem 1.2. The entire proof relies on a  $d$ -regular  $n$ -vertex graph  $G_1$ , such that (i) its girth is  $\Omega(\log_d n)$ ; and (ii) its adjacency matrix  $A_1$  has eigenvalues  $\lambda_1 \geq \dots \geq \lambda_n$  that satisfy  $\max\{|\lambda_2|, |\lambda_n|\} \leq \frac{1}{4}d^{2/3}$  (this bound is somewhat arbitrary, chosen to simplify the exposition). We actually need such a graph to exist for infinitely many  $n$ , with  $d$  bounded uniformly (as  $n$  grows). Such graphs are indeed known, for example the Ramanujan graphs constructed by Lubotzky, Philips and Sarnak [26] and by Margulis [27] for the case where  $d - 1$  is a prime, have eigenvalue upper bound  $2\sqrt{d-1}$  and girth lower bound  $(4/3 - o(1)) \log_{d-1} n$  (see e.g. [21]).

In what follows, let  $G_2$  be a certain isomorphic copy of  $G_1$  (i.e., obtained from  $G_1$  by permuting the vertices, as explained below). It will be convenient to assume that  $G_1$  and  $G_2$  have the *same* vertex set, which we denote by  $V$ , as then we can consider the multi-graph obtained by their edge union, denoted  $G_1 \cup G_2$ . Denoting the adjacency matrix of each  $G_i$  by  $A_i$ , the adjacency matrix of their edge union  $G_1 \cup G_2$  is simply  $A_1 + A_2$ . We can similarly view  $A_1 - A_2$  as the adjacency matrix of the same graph, except that now the edges are signed – those from  $G_1$  are positive, and those from  $G_2$  are negative.

The proof of the theorem will follow easily from the three propositions below. Proposition 4.1 provides combinatorial, girth-like, information about  $G_1 \cup G_2$ . Proposition 4.2 provides spectral information, like the condition number, about  $A_1 - A_2$ . These two propositions are proved by straightforward arguments, and the heart of the argument is in Proposition 4.3. This proposition constructs a PSD linear system based on  $A_1 - A_2$ , for which we can show that recovering a specific coordinate of the solution  $x$ , even approximately, requires many probes to  $b$ . Due to space constraints, the proof of the next proposition is omitted from this version.

► **Proposition 4.1.** *Let  $G_1$  be as above and fix a vertex  $\hat{w} \in V$ . Then there exists an isomorphic copy  $G_2$  of  $G_1$  (on the same vertex set), such that in their edge-union  $G_1 \cup G_2$ , the neighborhood of  $\hat{w}$  of radius  $r_{\text{tree}} \stackrel{\text{def}}{=} 0.2 \log_{4d} n$  is a  $2d$ -regular tree.*

► **Proposition 4.2.** *Let  $A_1, A_2$  be the adjacency matrices described above, and let  $\mu \stackrel{\text{def}}{=} 2\|A_1 - A_2\|$ . Then  $\mu \leq \frac{1}{2}d^{2/3}$ , and the matrix  $M \stackrel{\text{def}}{=} \mu I + A_1 - A_2 \in \mathbb{R}^{n \times n}$  is PSD with all its eigenvalues in the range  $[\frac{1}{2}\mu, \frac{3}{2}\mu]$ . Thus,  $M$  is invertible and has condition number  $\kappa(M) \leq 3$ .*

**Proof.** By the triangle inequality,  $\mu/2 = \|A_1 - A_2\| \leq \|A_1 - dI\| + \|(A_2 - dI)\| \leq 2 \max\{|\lambda_2|, |\lambda_n|\} \leq \frac{1}{2}d^{2/3}$ . The eigenvalues of  $A_1 - A_2$  are in the range  $[-\frac{1}{2}\mu, \frac{1}{2}\mu]$ , and thus those of  $M$  are in the range  $[\frac{1}{2}\mu, \frac{3}{2}\mu]$ . ◀

► **Proposition 4.3** (Proved in Section 4.1). *Let the graphs  $G_1, G_2$  be according to Proposition 4.1, let  $M \stackrel{\text{def}}{=} \mu I + A_1 - A_2 \in \mathbb{R}^{n \times n}$  as above, and fix  $r \leq r_{\text{tree}}/d^2$ . Then every randomized algorithm that, given input  $b \in \{-1, 0, +1\}^n$ , succeeds with probability at least  $6/7$  to approximate coordinate  $x_{\hat{w}}$  of  $x = M^{-1}b$  within additive error at most  $\frac{1}{5}\|x\|_\infty$ , must probe  $d^{\Omega(r)}$  entries from  $b \in \mathbb{R}^n$ , even when  $b$  is supported only on vertices at distance  $r$  from  $\hat{w}$  (in  $G_1 \cup G_2$ ).*

We can now prove Theorem 1.2 using the above 3 propositions. Let  $G_1, G_2, A_1, A_2$  and  $M$  be as required for these propositions, and fix  $r = r_{\text{tree}}/d^2$ . Let  $S \stackrel{\text{def}}{=} M$  and observe that it has the sparsity and condition number required for Theorem 1.2, and let the distinguished index be  $u \stackrel{\text{def}}{=} \hat{w}$ . Now consider a randomized algorithm that, given an input  $b \in \mathbb{R}^n$ , estimates coordinate  $x_u^*$  of  $x^* = S^{-1}b$ , or in other words, coordinate  $x_{\hat{w}}$  of  $x = M^{-1}b$ . We can then apply Proposition 4.3 and deduce that this algorithm must probe  $b \in \mathbb{R}^n$  in

$$d^{\Omega(r)} \geq d^{\Omega((\log_{4d} n)/d^2)} \geq n^{\Omega(1/d^2)}$$

entries, which proves Theorem 1.2.

#### 4.1 Proof of Proposition 4.3

Let  $V_k \subset V$  be the set of vertices at distance exactly  $k$  from  $\hat{w}$  in the edge-union graph  $G_1 \cup G_2$ . By Proposition 4.1, we can view the radius- $r_{\text{tree}}$  neighborhood of  $\hat{w}$  as a tree rooted at  $\hat{w}$ . In particular, for all  $k \leq r_{\text{tree}}$  we have  $|V_k| = 2d(2d-1)^{k-1}$ . For each vertex  $v \in V_k$ , let  $s_v \in \{\pm 1\}$  be the value of entry  $(\hat{w}, v)$  in  $(A_2 - A_1)^k$ , i.e., the product of the signs along the unique length- $k$  walk from  $\hat{w}$  to  $v$  in  $A_2 - A_1$  (i.e., the shortest path in  $G_1 \cup G_2$ ).

Now generate a random  $b \in \{-1, 0, +1\}^n$  as follows. First pick an arbitrary signal  $\sigma \in \{\pm 1\}$ ; then use it to choose for each  $v \in V_r$ , a random  $b_v \in \{\pm 1\}$  with a small bias  $\delta > 0$  (determined below) towards  $\sigma s_v \in \{\pm 1\}$ , i.e.,

$$\mathbb{E}[b_v | \sigma] = \left(\frac{1}{2} + \frac{\delta}{2}\right)\sigma s_v + \left(\frac{1}{2} - \frac{\delta}{2}\right)(-\sigma s_v) = \delta \sigma s_v.$$

Observe that  $\mathbb{E}[s_v b_v | \sigma] = s_v(\delta \sigma s_v) = \delta \sigma$ , which means that  $s_v b_v$  has a small bias towards the signal  $\sigma$ . Finally, let all other entries be 0, i.e.,  $b_v = 0$  for  $v \notin V_r$ . Observe that  $\|b\|_2^2 = |\text{supp}(b)| = |V_r|$  and  $\mathbb{E}[\sigma \sum_{v \in V_r} s_v b_v | \sigma] = \delta |V_r|$ . We set the bias to be

$$\delta \stackrel{\text{def}}{=} C(r^2 \log d) |V_r|^{-1/3} \tag{4}$$

for a sufficiently large constant  $C > 0$ . Notice that  $\{s_v\}_{v \in V_r}$  have fixed values known to the algorithm, hence observing  $b_v$  (by probing this entry of  $b$ ) is information-theoretically equivalent to observing  $s_v b_v$ .



The next lemma is standard and follows easily from Yao’s minimax principle, together with a bound on the total-variation distance between two Binomial distributions, with biases  $\frac{1}{2} + \delta$  and  $\frac{1}{2} - \delta$ , see e.g. [9, Fact D.1.3] or [1, Eqn. (2.15)].

► **Lemma 4.4.** *Every randomized algorithm that, with probability at least  $1/2 + \gamma$  for  $\gamma \in (0, 1/2)$ , recovers an unknown signal  $\sigma \in \{\pm 1\}$  from  $b_1, b_2, \dots \in \{\pm 1\}$ , each set independently to  $\sigma$  or  $-\sigma$  with bias  $\delta > 0$ , must probe at least  $\Omega(\delta^{-2}\gamma^2)$  entries of  $b$ .*

Thus, all probabilities from this point onward are computed over the randomness of  $b$ . We proceed to analyze  $x_{\hat{w}}$ , aiming to show that it can be used to recover  $\sigma$ , namely, that with high probability  $\text{sgn}(x_{\hat{w}}) = \sigma$ . Later we will bound  $\|x\|_\infty$  aiming to show a similar conclusion for  $x_{\hat{w}} \pm \frac{1}{5}\|x\|_\infty$ . For convenience, denote  $B \stackrel{\text{def}}{=} \frac{A_2 - A_1}{\mu}$ , hence  $\|B\| = \frac{\mu/2}{\mu} = \frac{1}{2}$  and

$$M^{-1} = (\mu(I - \frac{A_2 - A_1}{\mu}))^{-1} = \mu^{-1} \sum_{i \geq 0} B^i,$$

and since  $B$  is symmetric, for every vertex  $u \in V$  (including  $\hat{w}$ ),

$$x_u = \langle e_u, M^{-1}b \rangle = \mu^{-1} \sum_{i \geq 0} \langle e_u, B^i b \rangle = \mu^{-1} \sum_{i \geq 0} b^\top B^i e_u. \quad (5)$$

Each summand  $b^\top B^i e_u$  can be viewed as the summation, over all length- $i$  walks from vertex  $u$ , of the coordinate  $b_v$  corresponding to the walk’s end-vertex  $v$ , multiplied by  $\mu^{-i}$  and by the product of the signs of  $A_2 - A_1$  along the walk. We can restrict the summation to walks ending at vertices  $v \in V_r$ , as otherwise  $b_v = 0$ .

► **Lemma 4.5.** *For every vertex  $u \in V$  (including  $\hat{w}$ ),*

$$\sum_{i \geq 2r \log \mu} \left| b^\top B^i e_u \right| \leq \frac{1}{4} \mu^{-2r} \cdot \delta |V_r|.$$

**Proof of Lemma 4.5.** For each  $i$ , we have by Cauchy-Schwartz  $|b^\top B^i e_u| \leq \|b\|_2 \cdot \|B\|_2^i \leq |V_r|^{1/2} \cdot 2^{-i}$ , and then by our choice of the bias  $\delta$  in (4),

$$\sum_{i \geq 2r \log \mu} |b^\top B^i e_u| \leq |V_r|^{1/2} \sum_{i \geq 2r \log \mu} 2^{-i} \leq (|V_r| \cdot \delta / 8) \cdot 2\mu^{-2r}. \quad \blacktriangleleft$$

Recall that by Proposition 4.1, the neighborhood of  $\hat{w}$  of radius  $r_{\text{tree}}$  is a tree, and view it as a tree rooted at  $\hat{w}$ . For a vertex  $u$  in this tree, let  $S_u$  be the set of all vertices  $v \in V_r$  that are descendants of  $u$ ; for example,  $S_{\hat{w}} = V_r$ , and if the distance of  $u$  from  $\hat{w}$  is greater than  $r$  then  $S_u = \emptyset$ . Define a random variable  $Z_u \stackrel{\text{def}}{=} \sum_{v \in S_u} s_v b_v$ , whose expectation is

$$\mathbb{E}[Z_u] = \sum_{v \in S_u} \mathbb{E}[s_v b_v \mid \sigma] = |S_u| \cdot \delta \sigma.$$

► **Lemma 4.6.** *With probability at least  $6/7$ ,*

$$\forall 0 \leq k \leq r, \forall u \in V_k, \quad |Z_u - \mathbb{E}[Z_u]| \leq O\left(\sqrt{|S_u|} \cdot \ln(3|V_k|)\right). \quad (6)$$

We remark that the constant 3 is somewhat arbitrary but needed to make sure the righthand-side is positive even for  $k = 0$  (as  $|V_0| = 1$ ). In addition, applying (6) to  $\hat{w} \in V_0$  yields, by our choice of the bias  $\delta$  in (4),

$$|Z_{\hat{w}} - \mathbb{E}[Z_{\hat{w}}]| \leq O\left(\sqrt{|V_r|} \cdot \ln(3|V_r|)\right) \leq \frac{1}{4} \delta |V_r|. \quad (7)$$

### 3:16 On Solving Linear Systems in Sublinear Time

**Proof of Lemma 4.6.** Fix  $0 \leq k \leq r$  and  $u \in V_k$ . By Hoeffding's inequality, for every  $c > 0$ ,

$$\begin{aligned} \Pr \left[ |Z_u - \mathbb{E}[Z_u]| \geq c\sqrt{|S_u| \ln(3|V_k|)} \right] &\leq e^{-2c^2|S_u| \ln(3|V_k|)/(4|S_u|)} \\ &\leq e^{-(c^2/2) \ln(3|V_k|)} = (3|V_k|)^{-c^2/2}. \end{aligned}$$

By a union bound over all  $u \in V_0 \cup \dots \cup V_r$ ,

$$\Pr \left[ \exists u, |Z_u - \mathbb{E}[Z_u]| \geq c\sqrt{|S_u| \ln(3|V_k|)} \right] \leq \sum_{k=0}^r |V_k| \cdot (3|V_k|)^{-c^2/2} = \frac{1}{3} \sum_{k=0}^r (3|V_k|)^{1-c^2/2}.$$

For all  $c \geq 2$  this series is decreasing geometrically, because  $|V_k|$  grows at least by a factor of  $2d - 1 \geq 5$ , and thus the sum is dominated by its first term. By choosing  $c$  to be an appropriate constant, the first term (and the entire sum) can be made arbitrarily small. ◀

We assume henceforth that the event described in Lemma 4.6 occurs. Let  $W_i$  be the set of all walks of length  $i$  that start at  $\hat{w}$  and end (at some vertex) in  $V_r$ , i.e., at distance exactly  $r$  from  $\hat{w}$ . To simplify notation (later), define

$$Q \stackrel{\text{def}}{=} \sum_{i=r}^{5r \log \mu} \mu^{-i} |W_i|.$$

We make two remarks. First, we can equivalently start the summation from  $i = 0$ , because  $W_i = \emptyset$  for all  $i < r$ . Second, the range of  $i$  here complements the one in Lemma 4.5, except that the constant 5 here is intentionally bigger than the 2 there, creating a slack needed at the very end of the proof.

► **Lemma 4.7.** *If the event in Lemma 4.6 occurs, then*

$$x_{\hat{w}} \in (\sigma \pm \frac{1}{2})\delta \cdot \mu^{-1}Q,$$

and thus  $\text{sgn}(x_{\hat{w}}) = \sigma$  (i.e., recovers the signal).

**Proof of Lemma 4.7.** We would like to employ (5) and the interpretation of  $b^\top B^i e_{\hat{w}}$  via walks of length  $i$ . To this end, fix  $0 \leq i \leq 5r \log \mu$ . Observe that  $i \leq r_{\text{tree}}$ , hence a walk of length  $i$  from  $\hat{w}$  is entirely contained in the  $2d$ -regular tree formed by the neighborhood of  $\hat{w}$  of radius  $r_{\text{tree}}$ . Each such walk contributes to  $b^\top B^i e_{\hat{w}}$  the value  $b_v$  at the walk's end vertex  $v$ , multiplied by all the signs seen along the walk. We make two observations. First, we can restrict attention to end vertices  $v \in V_r$  (and in particular  $i \geq r$ ), because otherwise  $b_v = 0$  and the contribution is 0. Second, because it is a tree, every edge along the shortest path between  $\hat{w}$  and  $v$  (the start and end vertices) is traversed by the walk an odd number of times, and every other edge is traversed an even number of times. Hence, the product of the signs along the walk equals the product along that shortest path, which is exactly  $s_v$ . By symmetry, the number of walks ending at each  $v \in V_r$  is the same, namely,  $\frac{|W_i|}{|V_r|}$ , and thus

$$b^\top B^i e_{\hat{w}} = \sum_{v \in V_r} \frac{|W_i|}{|V_r|} \mu^{-i} s_v b_v = \frac{Z_{\hat{w}}}{|V_r|} \cdot \mu^{-i} |W_i|. \quad (8)$$

Assuming the event in Lemma 4.6 occurs, we have  $Z_{\hat{w}} \in (\delta\sigma|S_{\hat{w}}| \pm \frac{1}{4}\delta|V_r|) = (1 \pm \frac{1}{4})\sigma\delta|V_r|$ , and therefore (recall terms for  $i < r$  have zero contribution)

$$\sum_{i=0}^{5r \log \mu} b^\top B^i e_{\hat{w}} \in \sum_{i=r}^{5r \log \mu} (1 \pm \frac{1}{4})\sigma\delta \cdot \mu^{-i} |W_i| = (1 \pm \frac{1}{4})\sigma\delta \cdot Q.$$

For the range of  $i > 5r \log \mu$ , we can use Lemma 4.5 and the obvious  $|W_r| = |V_r|$  to derive

$$\left| \sum_{i > 5r \log \mu} b^\top B^i e_{\hat{w}} \right| \leq \sum_{i > 5r \log \mu} \left| b^\top B^i e_{\hat{w}} \right| \leq \frac{1}{4} \mu^{-2r} \cdot \delta |V_r| \leq \frac{1}{4} \delta Q.$$

Altogether, plugging into (5) we obtain

$$\mu \cdot x_{\hat{w}} = \sum_{i \geq 0} b^\top B^i e_{\hat{w}} \in \sum_{i=0}^{5r \log \mu} (1 \pm \frac{1}{4}) \sigma \delta \cdot Q \pm \frac{1}{4} \delta \cdot Q = (1 \pm \frac{1}{2}) \sigma \delta \cdot Q,$$

which proves the lemma because  $\sigma \in \{\pm 1\}$ . ◀

► **Lemma 4.8.** *If the event in Lemma 4.6 occurs, then*

$$\|x\|_\infty \leq 2\delta \cdot \mu^{-1} Q.$$

The proof of this lemma is omitted here but appears in the full version.

We can now complete the proof of Proposition 4.3. By Lemma 4.6, with probability at least  $6/7$  the event described therein occurs. Assume this is the case and consider an estimate  $\hat{x}_{\hat{w}}$  for  $x_{\hat{w}}$  that has additive error at most  $\epsilon \|x\|_\infty$  for  $\epsilon \leq \frac{1}{5}$ . By Lemma 4.7 we have  $x_{\hat{w}} \in (\sigma \pm \frac{1}{2}) \delta \cdot \mu^{-1} Q$ , and by Lemma 4.8 we have  $\|x\|_\infty \leq 2\delta \cdot \mu^{-1} Q$ . Altogether

$$\hat{x}_{\hat{w}} \in x_{\hat{w}} \pm \frac{1}{5} \|x\|_\infty \subseteq (\sigma \pm \frac{1}{2} \pm \frac{2}{5}) \delta \cdot \mu^{-1} Q,$$

which implies that  $\text{sgn}(x_{\hat{w}}) = \sigma$ .

Now consider a randomized algorithm for estimating  $x_{\hat{w}}$ , and whose output  $\hat{x}_{\hat{w}}$  satisfies the above additive bound with probability at least  $6/7$ . We can use this estimation algorithm to recover the signal  $\sigma$ , by simply reporting the sign of its estimate, namely  $\text{sgn}(x_{\hat{w}})$ . This recovery does not require additional probes to  $b$ , and by a union bound, it succeeds (in recovering  $\sigma$ ) with probability at least  $5/7$ . But by Lemma 4.4, such a recovery algorithm, and in particular the algorithm for estimating  $x_{\hat{w}}$ , must probe  $b$  in at least

$$\Omega(\delta^{-2}) \geq \Omega\left(|V_r|^{2/3} / (r^4 \log^2 d)\right) \geq \Omega\left((2d-1)^{2r/3} / (r^4 \log^2 d)\right) \geq d^{\Omega(r)}$$

entries, which proves Proposition 4.3.

---

## References

- 1 José A. Adell and P. Jodrá. Exact Kolmogorov and total variation distances between some familiar discrete distributions. *Journal of Inequalities and Applications*, 2006(1):64307, 2006. doi:10.1155/JIA/2006/64307.
- 2 Andris Ambainis. Variable time amplitude amplification and quantum algorithms for linear algebra problems. In *29th Symposium on Theoretical Aspects of Computer Science (STACS'12)*, volume 14, pages 636–647. LIPIcs, 2012. doi:10.4230/LIPIcs.STACS.2012.636.
- 3 Reid Andersen, Christian Borgs, Jennifer T. Chayes, John E. Hopcroft, Vahab S. Mirrokni, and Shang-Hua Teng. Local computation of PageRank contributions. *Internet Mathematics*, 5(1):23–45, 2008. doi:10.1080/15427951.2008.10129302.
- 4 Reid Andersen, Fan R. K. Chung, and Kevin J. Lang. Using PageRank to locally partition a graph. *Internet Mathematics*, 4(1):35–64, 2007. doi:10.1080/15427951.2007.10129139.
- 5 Reid Andersen and Yuval Peres. Finding sparse cuts locally using evolving sets. In *Proceedings of the Forty-first Annual ACM Symposium on Theory of Computing, STOC '09*, pages 235–244. ACM, 2009. doi:10.1145/1536414.1536449.

- 6 Christian Borgs, Michael Brautbar, Jennifer T. Chayes, and Shang-Hua Teng. Multiscale matrix sampling and sublinear-time PageRank computation. *Internet Mathematics*, 10(1-2):20–48, 2014. doi:10.1080/15427951.2013.802752.
- 7 Marco Bressan, Enoch Peserico, and Luca Pretto. Brief announcement: On approximating PageRank locally with sublinear query complexity. In *30th on Symposium on Parallelism in Algorithms and Architectures*, SPAA '18, pages 87–89. ACM, 2018. arXiv:1404.1864, doi:10.1145/3210377.3210664.
- 8 Sergey Brin and Lawrence Page. The anatomy of a large-scale hypertextual web search engine. *Computer Networks*, 30(1-7):107–117, 1998. doi:10.1016/S0169-7552(98)00110-X.
- 9 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. URL: <http://eccc.hpi-web.de/report/2015/063>.
- 10 Yen-Yu Chen, Qingqing Gan, and Torsten Suel. Local methods for estimating PageRank values. In *Proceedings of the Thirteenth ACM International Conference on Information and Knowledge Management*, CIKM '04, pages 381–389. ACM, 2004. doi:10.1145/1031171.1031248.
- 11 A. Childs, R. Kothari, and R. Somma. Quantum algorithm for systems of linear equations with exponentially improved dependence on precision. *SIAM Journal on Computing*, 46(6):1920–1950, 2017. doi:10.1137/16M1087072.
- 12 Fan Chung and Olivia Simpson. Solving local linear systems with boundary conditions using heat kernel PageRank. *Internet Mathematics*, 11(4-5):449–471, 2015. doi:10.1080/15427951.2015.1009522.
- 13 Fan Chung and Wenbo Zhao. *PageRank and Random Walks on Graphs*, pages 43–62. Springer, 2010. doi:10.1007/978-3-642-13580-4\_3.
- 14 Michael B. Cohen, Rasmus Kyng, Gary L. Miller, Jakub W. Pachocki, Richard Peng, Anup B. Rao, and Shen Chen Xu. Solving SDD linear systems in nearly  $m \log^{1/2} n$  time. In *Proceedings of the 46th Annual ACM Symposium on Theory of Computing*, pages 343–352, 2014. doi:10.1145/2591796.2591833.
- 15 Danial Dervovic, Mark Herbster, Peter Mountney, Simone Severini, Naïri Usher, and Leonard Wossnig. Quantum linear systems algorithms: a primer. *CoRR*, abs/1802.08227, 2018. arXiv:1802.08227.
- 16 Dean Doron, François Le Gall, and Amnon Ta-Shma. Probabilistic logarithmic-space algorithms for Laplacian solvers. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, APPROX/RANDOM 2017*, pages 41:1–41:20, 2017. doi:10.4230/LIPIcs.APPROX-RANDOM.2017.41.
- 17 Dean Doron, Amir Sarid, and Amnon Ta-Shma. On approximating the eigenvalues of stochastic matrices in probabilistic logspace. *Comput. Complex.*, 26(2):393–420, June 2017. doi:10.1007/s00037-016-0150-y.
- 18 George E Forsythe and Richard A Leibler. Matrix inversion by a Monte Carlo method. *Mathematics of Computation*, 4(31):127–129, 1950. doi:10.1090/S0025-5718-1950-0038138-X.
- 19 François Le Gall. Powers of tensors and fast matrix multiplication. In *International Symposium on Symbolic and Algebraic Computation, ISSAC '14*, pages 296–303, 2014. doi:10.1145/2608628.2608664.
- 20 Aram W. Harrow, Avinatan Hassidim, and Seth Lloyd. Quantum algorithm for linear systems of equations. *Phys. Rev. Lett.*, 103:150502, Oct 2009. doi:10.1103/PhysRevLett.103.150502.
- 21 Shlomo Hoory, Nathan Linial, and Avi Wigderson. Expander graphs and their applications. *Bull. Amer. Math. Soc.*, 43(4):439–561, 2006. doi:10.1090/S0273-0979-06-01126-8.

- 22 Iordanis Kerenidis and Anupam Prakash. Quantum recommendation systems. In *8th Innovations in Theoretical Computer Science Conference (ITCS' 17)*, volume 67 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 49:1–49:21. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2017. doi:10.4230/LIPIcs.ITCS.2017.49.
- 23 Bo'az Klartag and Oded Regev. Quantum one-way communication can be exponentially stronger than classical communication. In *43rd Annual ACM Symposium on Theory of Computing*, STOC '11, pages 31–40, 2011. doi:10.1145/1993636.1993642.
- 24 Rasmus Kyng, Yin Tat Lee, Richard Peng, Sushant Sachdeva, and Daniel A Spielman. Sparsified Cholesky and multigrid solvers for connection Laplacians. In *48th Annual ACM Symposium on Theory of Computing*, pages 842–850. ACM, 2016.
- 25 Yin Tat Lee. Probabilistic spectral sparsification in sublinear time. *CoRR*, abs/1401.0085, 2014. arXiv:1401.0085.
- 26 Alexander Lubotzky, Ralph Phillips, and Peter Sarnak. Ramanujan graphs. *Combinatorica*, 8(3):261–277, 1988. doi:10.1007/BF02126799.
- 27 G. A. Margulis. Explicit group-theoretic constructions of combinatorial schemes and their applications in the construction of expanders and concentrators. *Problemy Peredachi Informatsii*, 24(1):51–60, 1988.
- 28 Ran Raz. Exponential separation of quantum and classical communication complexity. In *31st Annual ACM Symposium on Theory of Computing*, STOC '99, pages 358–367, 1999. doi:10.1145/301250.301343.
- 29 Ronitt Rubinfeld, Gil Tamir, Shai Vardi, and Ning Xie. Fast local computation algorithms. In *Innovations in Computer Science - ICS 2010*, pages 223–238, 2011. URL: <http://conference.iis.tsinghua.edu.cn/ICS2011/content/papers/36.html>.
- 30 Nitin Shyamkumar, Siddhartha Banerjee, and Peter Lofgren. Sublinear estimation of a single element in sparse linear systems. In *54th Annual Allerton Conference on Communication, Control, and Computing*, Allerton 2016, pages 856–860, 2016. doi:10.1109/ALLERTON.2016.7852323.
- 31 D. A. Spielman and S.-H. Teng. Nearly-linear time algorithms for graph partitioning, graph sparsification, and solving linear systems. In *36th Annual ACM Symposium on Theory of Computing*, pages 81–90. ACM, 2004. doi:10.1145/1007352.1007372.
- 32 Daniel A. Spielman. Algorithms, graph theory, and linear equations in Laplacian matrices. In *Proceedings of the International Congress of Mathematicians*, volume 4, pages 2698–2722, 2010. doi:10.1142/9789814324359\_0164.
- 33 Daniel A. Spielman and Nikhil Srivastava. Graph sparsification by effective resistances. *SIAM J. Comput.*, 40(6):1913–1926, December 2011. doi:10.1137/080734029.
- 34 Jukka Suomela. Survey of local algorithms. *ACM Comput. Surv.*, 45(2):24:1–24:40, March 2013. doi:10.1145/2431211.2431223.
- 35 Ewin Tang. A quantum-inspired classical algorithm for recommendation systems. *CoRR*, abs/1807.04271, 2018. arXiv:1807.04271.
- 36 Nisheeth K. Vishnoi.  $Lx = b$ . *Foundations and Trends in Theoretical Computer Science*, 8(1–2):1–141, 2013. doi:10.1561/04000000054.
- 37 Alastair J. Walker. An efficient method for generating discrete random variables with general distributions. *ACM Trans. Math. Softw.*, 3(3):253–256, 1977. doi:10.1145/355744.355749.
- 38 Wolfgang R Wasow. A note on the inversion of matrices by random walks. *Mathematical Tables and Other Aids to Computation*, 6(38):78–81, 1952. doi:10.2307/2002546.



# Placing Conditional Disclosure of Secrets in the Communication Complexity Universe

Benny Applebaum<sup>1</sup>

Tel Aviv University, Tel Aviv, Israel  
<https://www.eng.tau.ac.il/~bennyap/>  
benny.applebaum@gmail.com

Prashant Nalini Vasudevan<sup>2</sup>

UC Berkeley, Berkeley, USA  
<http://people.eecs.berkeley.edu/~prashvas>  
prashvas@berkeley.edu

---

## Abstract

---

In the *conditional disclosure of secrets* (CDS) problem (Gertner et al., J. Comput. Syst. Sci., 2000) Alice and Bob, who hold  $n$ -bit inputs  $x$  and  $y$  respectively, wish to release a common secret  $z$  to Carol (who knows both  $x$  and  $y$ ) if and only if the input  $(x, y)$  satisfies some predefined predicate  $f$ . Alice and Bob are allowed to send a single message to Carol which may depend on their inputs and some shared randomness, and the goal is to minimize the communication complexity while providing information-theoretic security.

Despite the growing interest in this model, very few lower-bounds are known. In this paper, we relate the CDS complexity of a predicate  $f$  to its communication complexity under various communication games. For several basic predicates our results yield tight, or almost tight, lower-bounds of  $\Omega(n)$  or  $\Omega(n^{1-\epsilon})$ , providing an exponential improvement over previous logarithmic lower-bounds.

We also define new communication complexity classes that correspond to different variants of the CDS model and study the relations between them and their complements. Notably, we show that allowing for imperfect correctness can significantly reduce communication – a seemingly new phenomenon in the context of information-theoretic cryptography. Finally, our results show that proving explicit super-logarithmic lower-bounds for imperfect CDS protocols is a necessary step towards proving explicit lower-bounds against the class AM, or even  $AM \cap \text{coAM}$  – a well known open problem in the theory of communication complexity. Thus imperfect CDS forms a new minimal class which is placed just beyond the boundaries of the “civilized” part of the communication complexity world for which explicit lower-bounds are known.

**2012 ACM Subject Classification** Theory of computation → Communication complexity, Theory of computation → Cryptographic protocols

**Keywords and phrases** Conditional Disclosure of Secrets, Information-Theoretic Security

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2019.4

**Related Version** The full version of this paper is available as [6].

---

<sup>1</sup> Supported by the European Union’s Horizon 2020 Programme (ERC-StG-2014-2020) under grant agreement no. 639813 ERC-CLC, by an ICRC grant and by the Check Point Institute for Information Security.

<sup>2</sup> This work was done in part while the author was visiting Tel Aviv University. Supported in part by NSF Grant CNS-1350619, and by the Defense Advanced Research Projects Agency (DARPA) and the U.S. Army Research Office under contracts W911NF-15-C-0226 and W911NF-15-C-0236.



**Acknowledgements** We thank Prithvi Kamath and Hoeteck Wee for helpful discussions, and Mika Gos for helpful pointers. We also thank the anonymous reviewers for their useful comments.

## 1 Introduction

Understanding the communication complexity of information-theoretically secure protocols is a fundamental research problem. Despite much effort, we have very little understanding of the communication complexity of even simple cryptographic tasks, and for most models, there are exponentially large gaps between the best known upper-bounds and the best known lower-bounds. In an attempt to simplify the problem, one may try to focus on the most basic settings with a minimal non-trivial number of players (say two or three) and the simplest possible communication pattern (e.g., single message protocols). Different cryptographic tasks have been studied in this minimal setting, including secure computation [17], and non-interactive zero-knowledge proofs [23]. In this paper we will focus on what seems to be the simplest task in this model: *Conditional Disclosure of Secrets* (CDS) [20].<sup>3</sup>

### Conditional Disclosure of Secrets

Consider a pair of computationally unbounded parties, Alice and Bob, each holding an input,  $x \in \mathcal{X}$  and  $y \in \mathcal{Y}$  respectively, to some public predicate  $f : \mathcal{X} \times \mathcal{Y} \rightarrow \{0, 1\}$ . Alice and Bob also hold a joint secret  $z$  (say a single bit) and have access to a joint source of randomness  $r \stackrel{R}{\leftarrow} \mathcal{R}$ . The parties wish to disclose the secret  $z$  to a third party, Carol, if and only if the predicate  $f(x, y)$  evaluates to 1. To this end, Alice and Bob should each send a single message  $a = a(x, z; r)$  and  $b = b(y, z; r)$  to Carol. Based on the transcript  $(a, b)$  and the inputs  $(x, y)$ , Carol should be able to recover the secret  $z$  if and only if  $f(x, y) = 1$ . (Note that Carol is assumed to know  $x$  and  $y$ .) That is, we require two properties:

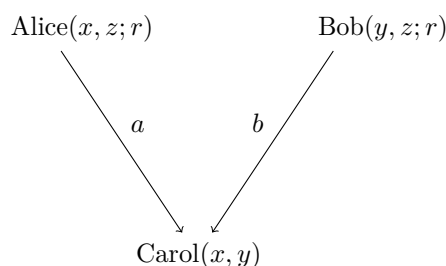
- *Correctness*: There exists a deterministic decoder algorithm  $\text{Dec}$  that recovers  $z$  from  $(x, y, a, b)$  with high probability whenever  $x, y$  is a 1-input (i.e.,  $f(x, y) = 1$ );
- *Privacy*: For every fixed 0-input  $(x, y)$  (for which the predicate evaluates to 0), regardless of the value of the secret  $z$ , the joint distribution of the transcript  $(a, b)$ , induced by a choice of the shared randomness, is statistically close (up to some small deviation error) to some canonical distribution  $\text{Sim}(x, y)$ .

The main complexity measure of CDS protocols is their communication complexity which is taken to be the total bit-length of the messages  $a$  and  $b$ . (See Figure 1 for a schematic view and Section A for formal definitions.)

Apart from being a natural basic notion, CDS has turned out to be a useful primitive with various applications in the context of private information retrieval (PIR) [20], secure multiparty computation [1, 25], secret sharing schemes [14, 15, 36, 31, 11, 2, 29], and attribute-based encryption [7, 37]. Correspondingly, the communication complexity of CDS was extensively studied in the last few years.

<sup>3</sup> While we do not wish to define the notions from [17] and [23], let us just mention that the complexity of a function in these two models upper-bounds the complexity in the CDS model [20, 5]. In this sense, CDS may be considered as being simpler.





■ **Figure 1** Schematic of a CDS protocol.

## Upper bounds

On the positive side, it is known that the CDS complexity of a predicate  $f$  is at most linear in the formula complexity of  $f$  [20]. This result was extended to other (presumably stronger) computational models such as (arithmetic) branching programs [26], and (arithmetic) span programs [5]. The latter paper also shows that the CDS complexity of  $f$  is at most linear in the complexity of any zero-information Arthur Merlin (ZAM) protocol for  $f$ . (The ZAM model, introduced by [23], adds a zero-knowledge property to the standard AM communication complexity model.)<sup>4</sup> In a recent breakthrough, Liu, Vaikuntanathan and Wee [30] showed that the CDS complexity of any predicate  $f : \{0, 1\}^n \times \{0, 1\}^n \rightarrow \{0, 1\}$  over  $n$ -bit inputs is at most  $2^{\tilde{O}(\sqrt{n})}$ , improving over the exponential upper-bound of  $O(2^{n/2})$  from [10]. Applebaum et al. [3] showed that when the secret is very long (exponential in the size of the domain of the predicate) the overhead per each bit of  $z$  can be reduced to  $O(n)$ ; a constant-rate solution (in which the total communication is  $O(|z|)$ ) was recently given in [2].

## The quest for lower bounds

On the lower-bound front much less is known. While we have tight lower bounds for restricted forms of CDS (e.g., when the computations are restricted to linear functions [19, 9, 12]), only few, relatively weak, lower-bounds are known for general CDS. It is important to note that an insecure solution to the problem has a communication cost of 1 bit! (Let Alice send the secret in the clear regardless of her input.) Hence, any super-constant lower-bound is, in a sense, non-trivial. Indeed, unlike the case of standard communication games for which communication lower-bounds are based on the correctness properties of the protocol, the challenge here is to somehow capture the additional cost of *privacy*.

The first super-constant lower-bound was proved by Gay, Kerenidis, and Wee [19].

► **Theorem 1** ([19]). *For every predicate  $f : \mathcal{X} \times \mathcal{Y} \rightarrow \{0, 1\}$ ,*

$$\text{CDS}(f) \geq \Omega(\log(\text{R}_{A \rightarrow B}(f) + \text{R}_{B \rightarrow A}(f))),$$

where  $\text{R}_{A \rightarrow B}(f)$  denotes the one-way randomized communication complexity of  $f$ , and  $\text{CDS}(f)$  denotes the minimal communication complexity of a CDS protocol for  $f$  with privacy and correctness error of 0.1.<sup>5</sup>

<sup>4</sup> The theorem of [5] actually relates the communication and randomness complexity of CDS for  $f$  to the randomness and communication complexity of a ZAM protocol for the complement of  $f$ . However, using our results in this paper one can conclude that the CDS communication of  $f$  is at most linear in the ZAM communication of  $f$ .

<sup>5</sup> The theorem was originally proved for perfect CDS, however, the proof generalizes to the imperfect case (see [3]).

For  $n$ -bits predicates, Theorem 1 leads, at best, to a logarithmic lower-bound of  $\Omega(\log n)$ . Applebaum *et al.* [3] showed that this bound is essentially tight: There are (partial) functions whose randomized communication complexity is exponentially larger than their CDS complexity. They also proved a linear  $n$ -bit lower-bound for a random (non-explicit)  $n$ -bit predicate  $f : \{0, 1\}^n \times \{0, 1\}^n \rightarrow \{0, 1\}$ . An explicit version of this result was proved by [4].

► **Theorem 2** ([4]). *For every non-degenerate predicate<sup>6</sup>  $f : \mathcal{X} \times \mathcal{Y} \rightarrow \{0, 1\}$  whose largest 0-monochromatic rectangle is of size at most  $L$ ,*

$$\text{pCDS}(f) \geq \log \frac{|f^{-1}(0)|}{L} - \log \frac{|\mathcal{X} \times \mathcal{Y}|}{|f^{-1}(0)|} - 1 = 2 \log |f^{-1}(0)| - \log |\mathcal{X}| - \log |\mathcal{Y}| - \log L - 1,$$

where  $\text{pCDS}(f)$  denotes the minimal communication complexity of a CDS protocol for  $f$  with perfect privacy and perfect correctness.

The theorem is effective for predicates whose communication matrix is rich in zeroes, and at the same time avoids large zero-monochromatic rectangles. In particular, for mod-2 inner product over  $n$ -bit inputs, we get a tight lower-bound of  $n - O(1)$  and for Set-Intersection a lower-bound of  $\Omega(n)$ . Unfortunately, the theorem is not robust to errors, leaving the imperfect CDS complexity of these predicates wide open. Moreover, for many basic predicates the theorem does not even give logarithmic bounds either due to the lack of many zeroes (e.g., the Not-Equal predicate) or due to the existence of huge zero-rectangles (e.g., the Greater-Than predicate).

## This paper

Theorems 1 and 2 provide a very partial picture, and fall short of proving meaningful and robust lower-bounds for many basic predicates, such as Not-equal, Greater-Than, Intersection, and Index.<sup>7</sup> We believe that a full understanding of these simple cases is necessary for the more ambitious goal of proving stronger lower bounds. Our goal in this paper is to remedy the situation by providing new lower-bound techniques. Specifically, we enrich our lower-bound toolbox by relating the CDS complexity of a function to its communication complexity under various communication games. Our results provide simple, yet effective, ways to leverage privacy to construct communication protocols. They lead to new lower-bounds for perfect and imperfect CDS protocols, and allow us to establish new results regarding the relations between different variants of the CDS model.

## 2 Our Contribution

### 2.1 Perfectly-correct CDS and coNP Games

Our first theorem relates the complexity of any perfectly-correct CDS protocol for  $f$  to the non-deterministic communication complexity of  $f$ 's complement.

► **Theorem 3.** *For every predicate  $f : \mathcal{X} \times \mathcal{Y} \rightarrow \{0, 1\}$ ,*

$$\text{pcCDS}(f) \geq \Omega(\text{coNP}(f)) - O(\log(n)),$$

where  $n$  denotes the total input length of  $f$ , and  $\text{pcCDS}(f)$  denotes the minimal communication complexity of a CDS protocol for  $f$  with perfect correctness and privacy error of 0.1.

<sup>6</sup> A predicate is non-degenerate if for every fixing of  $x \in \mathcal{X}$  the residual function  $f(x, \cdot)$  is not the constant zero function.

<sup>7</sup> Apart of being basic examples, these predicates are motivated by some of the applications of CDS.

**Proof idea**

To prove the theorem, we first show that the coNP complexity is upper-bounded by the randomness complexity of the CDS, and then prove that one can always assume that the randomness complexity is comparable to the communication complexity via a new sparsification lemma (similar to that of Newman [33]). The first part relies on the following simple observation: In order to convince Alice and Bob that  $f(x, y)$  evaluates to zero it suffices to prove that the joint distribution of the CDS messages for zero-secret,  $(a(x, z = 0; r), b(y, z = 0; r))$ , induced by a random choice of  $r$ , and the joint distribution of the messages for one-secret  $(a(x, z = 1; r), b(y, z = 1; r))$ , are *not disjoint*. A prover can prove this statement by sending to Alice and Bob a pair of strings  $r_0$  and  $r_1$  for which  $(a(x, z = 0; r_0), b(y, z = 0; r_0))$  equals to  $(a(x, z = 1; r_1), b(y, z = 1; r_1))$ . (See full version [6] for details.)

Despite its simplicity, this theorem is quite powerful. In particular, ignoring the constants in the Omega-notation and the logarithmic loss, the bound provided by Theorem 3 subsumes the lower-bound of Theorem 2 from [4]. Indeed, the latter lower-bound is at most the logarithm of the ratio between the zero-mass of  $f$  and its largest zero-monochromatic rectangle – a quantity that cannot be larger than the non-deterministic communication complexity of the complement of  $f$  (i.e.,  $\text{coNP}(f)$ ). Moreover, our new theorem can be applied to predicates that have only few zero entries or to predicates with huge zero-rectangles, for which Theorem 2 becomes meaningless. For example, by plugging-in classical coNP lower-bounds, we settle the complexity of the not-equal predicate with respect to perfectly correct CDS protocols.

► **Corollary 4.** *Let  $\text{NEQ}_n : \{0, 1\}^n \times \{0, 1\}^n \rightarrow \{0, 1\}$  denote the not-equal predicate which evaluates to 1 if and only if  $x \neq y$ . Then,*

$$\text{pcCDS}(\text{NEQ}_n) \geq \Omega(n).$$

Similar tight linear lower-bounds can be obtained for the pcCDS complexity of the Greater-Than predicate, the Set-Intersection predicate, and the Inner-Product predicate. Previously, we had no super-logarithmic lower bounds that tolerate privacy error. (As already mentioned, for Greater-Than and  $\text{NEQ}_n$ , we did not have such bounds even for perfect CDS protocols.)

**pcCDS is not closed under complement**

Interestingly, the *equality* function  $\text{EQ}_n$  has a very succinct perfect CDS protocol: Use the shared randomness to sample a pair-wise independent hash function  $h : \{0, 1\}^n \rightarrow \{0, 1\}$ , and let Alice output  $h(x)$  and Bob output  $h(y) \oplus z$ . The protocol has a minimal communication complexity of 2 and randomness complexity of  $O(n)$ . (The latter can be reduced to  $O(\log n)$  by using an almost pair-wise independent hash function and settling for a constant privacy error.) This yields a strong separation between the complexity of a predicate and its complement with respect to perfectly-correct perfectly-private CDS protocols (pCDS).

► **Corollary 5.**  *$\text{pCDS}(\text{EQ}_n) = 2$  whereas  $\text{pCDS}(\text{NEQ}_n) \geq \text{pcCDS}(\text{NEQ}_n) \geq \Omega(n)$ . In particular, the classes pCDS and pcCDS are not closed under complement.<sup>8</sup>*

Transformations from CDS protocols for  $f$  to its complement were studied in [3]. The resulting protocols either introduce a privacy error or suffer from a communication overhead that grows polynomially with the randomness complexity of the original protocol. The  $\text{NEQ}_n$  example shows that at least one of these losses is inherent.

<sup>8</sup> We follow the standard communication complexity terminology and write pCDS to denote the class of predicates that admit a pCDS protocol whose complexity is polylogarithmic in the input length. A similar convention will be used throughout the paper for all other variants of the CDS model.

### The benefit of decoding errors

The results of [3] (together with our randomness sparsification lemma) show that imperfect CDS is closed under complement. This general result leads to a polylogarithmic CDS protocol for  $\text{NEQ}_n$  with imperfect privacy and imperfect correctness, providing a surprising separation between general imperfect CDS protocols and ones which have perfect correctness. In fact, it is not hard to directly design a CDS protocol for  $\text{NEQ}_n$  with *constant communication*, *perfect privacy*, and constant correctness error. (See the full version [6] for a more general statement.) This leads to the following stronger separation.

► **Corollary 6.** *There is an  $n$ -bit predicate  $f$  for which  $\text{pcCDS}(f) = \Omega(n)$  and  $\text{ppCDS}(f) = O(1)$ , where  $\text{ppCDS}(f)$  denotes the minimal communication complexity of a CDS protocol for  $f$  with perfect privacy and correctness error of 0.1. In particular,*

$$\text{ppCDS} \not\subseteq \text{pcCDS}.$$

As pointed to us by Hoteck Wee, Corollary 6 provides a rare example for an information-theoretic secure protocol that can significantly benefit from a small correctness error. This phenomena seems new in the context of information-theoretic secure cryptography, and is worth further exploration.<sup>9</sup>

## 2.2 Perfectly-Private CDS and PP Games

Our next goal is to lower-bound the complexity of CDS protocols with correctness errors. We begin with the case of perfectly private protocols.

► **Theorem 7.** *For every predicate  $f : \mathcal{X} \times \mathcal{Y} \rightarrow \{0, 1\}$ ,*

$$\text{ppCDS}(f) \geq \Omega(\text{PP}(f)) - O(\log(n)),$$

where  $n$  denotes the total input length of  $f$ , and  $\text{ppCDS}(f)$  denotes the minimal communication complexity of a CDS protocol for  $f$  with perfect privacy and correctness error of 0.1.

The complexity measure  $\text{PP}(f)$  essentially corresponds to the sum of the communication complexity and number of private random bits used by a communication protocol that computes  $f$  correctly with probability more than  $1/2$ , where shared randomness is not allowed. (See the full version [6] for a formal definition.) The discrepancy method implies that the PP complexity of the mod-2 inner-product predicate  $\text{IP}_n$  is  $\Omega(n)$  (cf. [28]) and so we get the following.

► **Corollary 8.** *Let  $\text{IP}_n : \{0, 1\}^n \times \{0, 1\}^n \rightarrow \{0, 1\}$  denote the inner-product predicate on  $n$ -bit inputs. Then,*

$$\text{ppCDS}(\text{IP}_n) \geq \Omega(n).$$

This is the first linear lower-bound on CDS with imperfect correctness. (Previous arguments fail to achieve such a result even for a non-explicit predicate.)

---

<sup>9</sup> Compare this, for example, to Shannon's classical lower-bound for perfectly-secure one-time symmetric encryption [35] in which a constant decryption error has a minor effect on the key/ciphertext length [16].

**Proof idea**

In order to prove Theorem 7, we turn a ppCDS protocol into a PP protocol. Loosely speaking, the idea is to construct a randomized protocol that accepts the input  $(x, y)$  based on collisions between random CDS transcripts that correspond to a zero-secret and random CDS transcripts that correspond to a one-secret. This idea, which was employed in the query setting by [13], leads to the desired result. (Details appear in the full version [6].)

**2.3 Imperfect CDS, Interactive Proofs, and Zero Knowledge**

We move on to the most general case of imperfect CDS protocols with both constant privacy error and correctness error. We show that the complexity of such protocols is at least polynomial in the AM communication complexity of  $f$ . (The latter class is the communication complexity analogue of Arthur-Merlin proofs.)

► **Theorem 9.** *There exists some universal constant  $c > 0$ , such that for any Boolean function  $f$  it holds that*

$$\text{CDS}(f) \geq \text{AM}(f)^c - \text{polylog}(n),$$

where  $n$  denotes the total input length of  $f$ , and  $\text{CDS}(f)$  denotes the minimal communication complexity of a CDS protocol for  $f$  with correctness and privacy errors of 0.1.

Since (imperfect) CDS is closed under complement (by [3, Theorem 2] and [6, Lemma 1]), it holds that  $\text{CDS}(f) \leq \text{poly}(\text{CDS}(f))$ , and so we conclude the following.

► **Corollary 10.** *There exists some universal constant  $c > 0$ , such that for any Boolean function  $f$  it holds that*

$$\text{CDS}(f) \geq \max(\text{AM}(f), \text{coAM}(f))^c - \text{polylog}(n),$$

where  $n$  denotes the total input length of  $f$ .

**Explicit CDS lower-bounds?**

Corollary 10 can be used to show that the CDS complexity of most  $n$ -bit predicates must be at least polynomial in  $n$ , even when the protocol is imperfect. Unfortunately, it falls short of providing explicit lower-bounds; Finding an explicit function outside  $\text{AM} \cap \text{coAM}$  is a central open problem in the theory of communication complexity. In fact,  $\text{AM} \cap \text{coAM}$  forms a minimal class for which no explicit lower-bounds are known [24]. Corollary 10 places CDS as a weaker (and perhaps more accessible) target for explicit lower-bounds.

**Proof idea**

To prove Theorem 9 we show that a CDS protocol can be transformed into a constant-round private-coins interactive-proof. Then, we note that, just like in the computational setting, such interactive proofs can be converted to an AM protocol with polynomial overhead [8, 22].<sup>10</sup> The first step is obtained by imitating the standard interactive proof of Graph Nonisomorphism [21]. Indeed, the AM protocol constructed in Theorem 9 turns out to satisfy a *statistical zero-knowledge* property; That is, the view of Alice and Bob can be simulated via a low complexity 2-party randomized protocol. (See the full version [6] for details.)

<sup>10</sup>This reduction has a polynomial dependency in the randomness. In order to avoid such an overhead in the final statement, we prove a randomness sparsification lemma for constant-round interactive protocols. This requires some care due to the use of private coins.

### CDS vs. SZK

Recall that, by definition, a CDS protocol yields a (distributed mapping) from the input  $(x, y)$  and the secret  $z$  to a distribution  $D_z$  over the transcript  $(a, b)$  such that the distributions,  $D_0$  and  $D_1$ , are either statistically-close or statistically-far depending on the value of  $f(x, y)$ . This resembles the Statistical Difference problem [34], which is known to be complete for the computational complexity class SZK (consisting of problems that have interactive proofs that are statistically zero-knowledge). One may therefore hope to prove that in the communication complexity setting CDS complexity is characterized by SZK complexity. As already mentioned, Theorem 9 actually shows that  $\text{CDS} \subseteq \text{SZK}$ , however, we do not know whether the reverse direction holds. Roughly speaking, such a result faces two obstacles. Firstly, the completeness result from [34] has an overhead that depends on the randomness complexity of the protocol, and we do not know how to get rid of this dependency. (In particular, it is not clear how to prove a proper sparsification lemma for SZK without sacrificing the zero-knowledge property.) Secondly, even if the randomness complexity is small, we do not know how to obtain a CDS protocol without allowing some interaction between Alice and Bob. Indeed, in the full version [6] we show that  $\text{SZK}' \subseteq \text{CDS}'$  where the “prime” version of SZK charges randomness towards the total complexity and the “prime” version of CDS allows short interaction between Alice and Bob. The problem of proving that  $\text{SZK} \subseteq \text{CDS}$  (and therefore  $\text{SZK} = \text{CDS}$ ) remains as an interesting open problem.

The results described so far are summarised in Figure 2, which shows the relationship between perfect and imperfect CDS and various measures from communication complexity. In Table 1, we list the current state of knowledge of the various CDS complexities of a number of commonly studied predicates. (See Section 3.)

## 2.4 Asymmetry in CDS and One-Way Communication

We shift gears, and turn to study the communication tradeoffs between Alice’s and Bob’s messages. Suppose that Alice’s message is restricted to a short string of length  $t_A$ . Can we prove that Bob’s message must be very long? We prove such tradeoffs based on the one-way randomized communication complexity of  $f$ .

► **Theorem 11.** *In any perfectly correct 0.1-private CDS protocol for  $f$  in which Alice and Bob communicate  $t_A$  and  $t_B$  bits respectively and the total input length of the function is  $n$ , it holds that:*

$$2^{t_A}(t_A + t_B + \log n) \geq \Omega(R_{B \rightarrow A}(f)).$$

(In fact, the result holds even if one considers one-way randomized protocols that err only over zero inputs.) Recall that Theorem 1 (which is from [19]) shows that the total communication complexity  $t_A + t_B$  is at least logarithmic in  $(R_{A \rightarrow B}(f) + R_{B \rightarrow A}(f))$ , which is tight for some predicates [3]. Theorem 11 provides a more accurate picture. If the total communication complexity is dominated by  $t_A$ , then one gets a logarithmic bound, similar to Theorem 1; however, when  $t_A$  is small (e.g., constant), we get a strong linear lower-bound of

$$t_B = \Omega(R_{B \rightarrow A}(f)) - O(\log n).$$

In fact, when  $R_{B \rightarrow A}(f) = \Omega(n)$ , for any constant  $\alpha < 1$  if  $t_A \leq \alpha \log n$  then

$$t_B = \Omega(n^{1-\alpha}).$$

Concretely, consider the  $\text{Index}_n$  predicate in which Bob holds an  $n$ -bit database  $x \in \{0, 1\}^n$  and Alice holds an index  $i \in [n]$  (encoded as a string of length  $\log n$ ) and the output is the  $i$ -th bit of  $x$ . Since  $R_{B \rightarrow A}(\text{Index}_n) = \Omega(n)$  [27] we get:

► **Corollary 12.** *In any perfectly correct 0.1-private CDS protocol for  $\text{Index}_n$  in which Alice communicates at most  $\alpha \log n + O(1)$  bits for some constant  $0 \leq \alpha < 1$ , the database owner, Bob, must communicate at least  $\Omega(n^{1-\alpha})$  bits.*

Similar results can be obtained for predicates like Greater-Than, Set-Disjointness and Set-Intersection, based on classical lower-bounds for randomized one-way communication complexity (cf. [32, 27]).

The  $\text{Index}_n$  predicate plays an important role in CDS constructions and applications. First, it is complete for CDS in the sense that any  $n$ -bit predicate can be reduced to  $\text{Index}_N$  for  $N = 2^n$ . Indeed, the best known general CDS protocols were obtained by improving the pCDS complexity of  $\text{Index}$  [30]. In addition, a CDS for the index function can be viewed as a one-time version of the well-studied notion of *Broadcast Encryption*, and the lower-bound of Corollary 12 becomes especially appealing under this framework. Details follow.

### Broadcast Encryption [18]

Suppose that we have a single sender and  $n$  receivers. The sender has a private encryption key  $r$  and each receiver  $i \in [n]$  has its own private decryption key  $k_i$ . All the keys were collectively generated and distributed in an offline phase. In an online phase, the sender gets a message  $z$  together with a public list of authorized users  $y \subseteq [n]$ , represented by an  $n$ -bit characteristic vector. The sender should broadcast a ciphertext  $b = b(y, z; r)$  to all the receivers (who also know  $y$ ) so that an *authorized* receiver will be able to decrypt the ciphertext, and an unauthorized (computationally unbounded) receiver will learn nothing about the message  $z$ . The goal is to minimize the length of the ciphertext  $b$ , and the length of the keys  $k_i$ .

Information-theoretic one-time secure Broadcast Encryption turns to be equivalent to the CDS problem with respect to the  $\text{Index}_n$  predicate: Identify the ciphertext with Bob's message  $b = b(y, z; r)$  and the  $i$ -th key with Alice's message  $a(i; r)$ .<sup>11</sup> The problem can be solved with  $n$ -bit ciphertext and 1-bit keys, and with 1-bit ciphertext and  $n$ -bit keys. In fact, [19] showed that one can smoothly get any tradeoff as long as the product of the ciphertext length and the key length is  $n$ . Corollary 12 shows that when the key-length is sub-logarithmic the ciphertext must be almost linear, confirming a conjecture of Wee [38].

### Proof idea (of Theorem 11)

The idea is to let Bob send to Alice a pair of random strings  $r_0$  and  $r_1$  that are mapped to the same Bob's message  $b$  under the zero-secret and under the one-secret respectively. Alice then uses the string  $r_z$  and the secret  $z$  to compute a corresponding message  $a_z$ , and accepts if the zero message  $a_0$  equals to the one message  $a_1$ . Perfect correctness guarantees that Alice will never err on 0-inputs. We further show that, when  $f(x, y) = 1$ , Alice accepts with probability which is at least inverse-exponential in her message length (up to a loss that is proportional to the privacy error of the protocol). See the full version [6] for details.

<sup>11</sup> Here we assume that we have a CDS in which only Bob holds the secret. However, any CDS can be transformed into this form with an additional communication cost of  $O(|z|) = O(1)$ .

■ **Table 1** The CDS complexity of some simple functions. By definition, an upper-bound in the leftmost column (pCDS) implies an upper-bound in all other columns, and a lower-bound in the rightmost column (CDS) implies a lower-bound in all other columns. All the linear upper-bounds for pCDS follow from the fact that all of these predicates can be computed by a linear-size formula. The logarithmic lower-bounds for CDS follow from Theorem 1 (and the fact that the corresponding predicates have linear randomized one-way communication complexity.) The linear lower-bounds for pcCDS and ppCDS follow from Theorems 3 and 7 respectively.

Predicate	pCDS	pcCDS	ppCDS	CDS
Equality	$\Theta(1)$	$\Theta(1)$	$\Theta(1)$	$\Theta(1)$
Non-Equality	$\Theta(n)$	$\Theta(n)$	$\Theta(1)$	$\Theta(1)$
Inner-Product	$\Theta(n)$	$\Theta(n)$	$\Theta(n)$	$O(n) \ \& \ \Omega(\log n)$
Greater-Than	$\Theta(n)$	$\Theta(n)$	$O(n) \ \& \ \Omega(\log n)$	$O(n) \ \& \ \Omega(\log n)$
Set-Intersection	$\Theta(n)$	$\Theta(n)$	$O(n) \ \& \ \Omega(\log n)$	$O(n) \ \& \ \Omega(\log n)$
Set-Disjointness	$O(n) \ \& \ \Omega(\log n)$	$O(n) \ \& \ \Omega(\log n)$	$O(n) \ \& \ \Omega(\log n)$	$O(n) \ \& \ \Omega(\log n)$

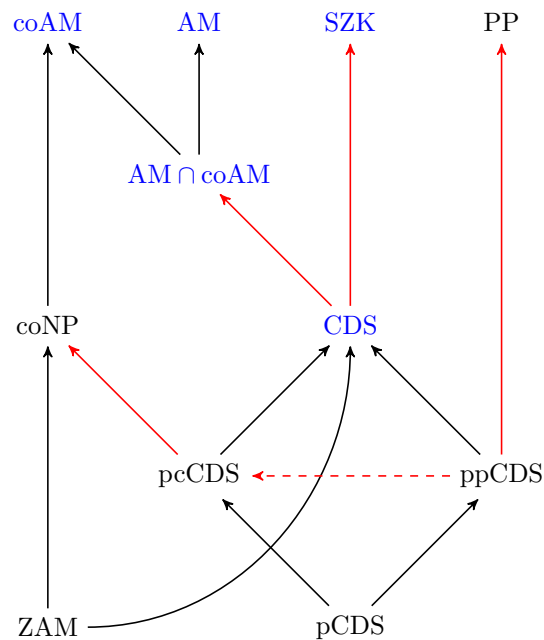
### 3 Conclusion and Open Questions

In this paper we studied the relations between CDS protocols and standard communication complexity games. We established new connections between CDS communication complexity (with perfect and imperfect privacy and correctness) to well-known communication complexity measures for non-deterministic protocols, randomized unbounded-error protocols, and one-way protocols. This leads to new CDS bounds for various simple functions. These results are summarized in Figure 2 and Table 1.

We end by listing the immediate interesting questions left open following our work.

1. Prove an explicit polynomial lower-bound on (imperfect) CDS complexity. (A natural candidate would be Inner-Product.)
2. Our current ppCDS lower-bounds are based on PP complexity, which corresponds to discrepancy. Can we derive such bounds on weaker, easier-to-establish, properties? In particular, can we prove non-trivial ppCDS lower-bounds for predicates that have low randomized bounded-error communication complexity like Greater-Than?
3. Unlike all the other communication complexity measures considered here, CDS complexity is not necessarily upper-bounded by the length of the inputs. But we have no super-linear (or even linear with a large constant factor) lower-bounds for even perfect CDS protocols. Can any of the existing lower-bound techniques from communication complexity be used to obtain such bounds?
4. If not, can this difficulty be explained, perhaps by relating the problem of proving such lower bounds for CDS to more well-studied problems that are still unsolved?
5. Following the paradigm of lifting query complexity lower bounds to the communication setting, is there a natural query complexity measure that can be lifted to CDS complexity?
6. One simple predicate that has eluded all our bounds is Set-Disjointness, for which the best (imperfect) CDS protocol we know has  $O(n)$  complexity, and the best lower bound we can prove, even for perfect CDS, is  $\Omega(\log(n))$ . Are either of these tight?





■ **Figure 2** As is standard, we use the name of a complexity measure to also denote the class of functions with  $\text{polylog}(n)$  complexity under the measure. For classes  $C_1$  and  $C_2$ , a solid arrow  $C_1 \rightarrow C_2$  indicates that  $C_1 \subseteq C_2$ , and a dashed arrow  $C_1 \dashrightarrow C_2$  indicates that  $C_1 \not\subseteq C_2$ . Red arrows indicate new results from this paper. Blue text indicates classes for which explicit bounds are not known.

---

## References

- 1 William Aiello, Yuval Ishai, and Omer Reingold. Priced Oblivious Transfer: How to Sell Digital Goods. In Birgit Pfitzmann, editor, *Advances in Cryptology - EUROCRYPT 2001, International Conference on the Theory and Application of Cryptographic Techniques, Innsbruck, Austria, May 6-10, 2001, Proceeding*, volume 2045 of *Lecture Notes in Computer Science*, pages 119–135. Springer, 2001. doi:10.1007/3-540-44987-6\_8.
- 2 Benny Applebaum and Barak Arkis. On the Power of Amortization in Secret Sharing: d-Uniform Secret Sharing and CDS with Constant Information Rate. In Amos Beimel and Stefan Dziembowski, editors, *Theory of Cryptography - 16th International Conference, TCC 2018, Panaji, India, November 11-14, 2018, Proceedings, Part I*, volume 11239 of *Lecture Notes in Computer Science*, pages 317–344. Springer, 2018. doi:10.1007/978-3-030-03807-6\_12.
- 3 Benny Applebaum, Barak Arkis, Pavel Raykov, and Prashant Nalini Vasudevan. Conditional Disclosure of Secrets: Amplification, Closure, Amortization, Lower-Bounds, and Separations. In Jonathan Katz and Hovav Shacham, editors, *Advances in Cryptology - CRYPTO 2017 - 37th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 20-24, 2017, Proceedings, Part I*, volume 10401 of *Lecture Notes in Computer Science*, pages 727–757. Springer, 2017. doi:10.1007/978-3-319-63688-7\_24.
- 4 Benny Applebaum, Thomas Holenstein, Manoj Mishra, and Ofer Shayevitz. The Communication Complexity of Private Simultaneous Messages, Revisited. In Jesper Buus Nielsen and Vincent Rijmen, editors, *Advances in Cryptology - EUROCRYPT 2018 - 37th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Tel Aviv, Israel, April 29 - May 3, 2018 Proceedings, Part II*, volume 10821 of *Lecture Notes in Computer Science*, pages 261–286. Springer, 2018. doi:10.1007/978-3-319-78375-8\_9.

- 5 Benny Applebaum and Pavel Raykov. From Private Simultaneous Messages to Zero-Information Arthur-Merlin Protocols and Back. In *Theory of Cryptography - 13th International Conference, TCC 2016-A, Tel Aviv, Israel, January 10-13, 2016, Proceedings, Part II*, pages 65–82, 2016. doi:10.1007/978-3-662-49099-0\_3.
- 6 Benny Applebaum and Prashant Nalini Vasudevan. Placing conditional disclosure of secrets in the communication complexity universe. *Technical Report*, 2018. Full version of this paper. URL: <https://www.eng.tau.ac.il/~bennyap/publications.html>.
- 7 Nuttapong Attrapadung. Dual System Encryption via Doubly Selective Security: Framework, Fully Secure Functional Encryption for Regular Languages, and More. In Phong Q. Nguyen and Elisabeth Oswald, editors, *Advances in Cryptology - EUROCRYPT 2014 - 33rd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Copenhagen, Denmark, May 11-15, 2014. Proceedings*, volume 8441 of *Lecture Notes in Computer Science*, pages 557–577. Springer, 2014. doi:10.1007/978-3-642-55220-5\_31.
- 8 László Babai. Trading Group Theory for Randomness. In Robert Sedgewick, editor, *Proceedings of the 17th Annual ACM Symposium on Theory of Computing, May 6-8, 1985, Providence, Rhode Island, USA*, pages 421–429. ACM, 1985. doi:10.1145/22145.22192.
- 9 Amos Beimel, Oriol Farràs, Yuval Mintz, and Naty Peter. Linear Secret-Sharing Schemes for Forbidden Graph Access Structures. To appear in TCC 2017, 2017.
- 10 Amos Beimel, Yuval Ishai, Ranjit Kumaresan, and Eyal Kushilevitz. On the Cryptographic Complexity of the Worst Functions. In Yehuda Lindell, editor, *Theory of Cryptography - 11th Theory of Cryptography Conference, TCC 2014, San Diego, CA, USA, February 24-26, 2014. Proceedings*, volume 8349 of *Lecture Notes in Computer Science*, pages 317–342. Springer, 2014. doi:10.1007/978-3-642-54242-8\_14.
- 11 Amos Beimel, Eyal Kushilevitz, and Pnina Nissim. The Complexity of Multiparty PSM Protocols and Related Models. To appear in Eurocrypt 2018, 2018. URL: <https://eprint.iacr.org/2018/148>.
- 12 Amos Beimel and Naty Peter. Optimal Linear Multiparty Conditional Disclosure of Secrets Protocols. *Cryptology ePrint Archive*, Report 2018/441, 2018. URL: <https://eprint.iacr.org/2018/441>.
- 13 Adam Bouland, Lijie Chen, Dhiraj Holden, Justin Thaler, and Prashant Nalini Vasudevan. On the Power of Statistical Zero Knowledge. In Chris Umans, editor, *58th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2017, Berkeley, CA, USA, October 15-17, 2017*, pages 708–719. IEEE Computer Society, 2017. doi:10.1109/FOCS.2017.71.
- 14 Ernest F. Brickell and Daniel M. Davenport. On the Classification of Ideal Secret Sharing Schemes. *J. Cryptology*, 4(2):123–134, 1991. doi:10.1007/BF00196772.
- 15 Renato M. Capocelli, Alfredo De Santis, Luisa Gargano, and Ugo Vaccaro. On the Size of Shares for Secret Sharing Schemes. *J. Cryptology*, 6(3):157–167, 1993. doi:10.1007/BF00198463.
- 16 Yevgeniy Dodis. Shannon Impossibility, Revisited. In Adam D. Smith, editor, *Information Theoretic Security - 6th International Conference, ICITS 2012, Montreal, QC, Canada, August 15-17, 2012. Proceedings*, volume 7412 of *Lecture Notes in Computer Science*, pages 100–110. Springer, 2012. doi:10.1007/978-3-642-32284-6\_6.
- 17 Uriel Feige, Joe Kilian, and Moni Naor. A minimal model for secure computation (extended abstract). 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 554–563. ACM, 1994. doi:10.1145/195058.195408.
- 18 Amos Fiat and Moni Naor. Broadcast Encryption. In Douglas R. Stinson, editor, *Advances in Cryptology - CRYPTO '93, 13th Annual International Cryptology Conference, Santa Barbara, California, USA, August 22-26, 1993, Proceedings*, volume 773 of *Lecture Notes in Computer Science*, pages 480–491. Springer, 1993. doi:10.1007/3-540-48329-2\_40.

- 19 Romain Gay, Iordanis Kerenidis, and Hoeteck Wee. Communication Complexity of Conditional Disclosure of Secrets and Attribute-Based Encryption. In Rosario Gennaro and Matthew Robshaw, editors, *Advances in Cryptology - CRYPTO 2015 - 35th Annual Cryptology Conference, Santa Barbara, CA, USA, August 16-20, 2015, Proceedings, Part II*, volume 9216 of *Lecture Notes in Computer Science*, pages 485–502. Springer, 2015. doi:10.1007/978-3-662-48000-7\_24.
- 20 Yael Gertner, Yuval Ishai, Eyal Kushilevitz, and Tal Malkin. Protecting Data Privacy in Private Information Retrieval Schemes. *J. Comput. Syst. Sci.*, 60(3):592–629, 2000. doi:10.1006/jcss.1999.1689.
- 21 Oded Goldreich, Silvio Micali, and Avi Wigderson. Proofs that Yield Nothing But Their Validity for All Languages in NP Have Zero-Knowledge Proof Systems. *J. ACM*, 38(3):691–729, 1991. doi:10.1145/116825.116852.
- 22 Shafi Goldwasser and Michael Sipser. Private Coins versus Public Coins in Interactive Proof Systems. *Advances in Computing Research*, 5:73–90, 1989.
- 23 Mika Göös, Toniann Pitassi, and Thomas Watson. Zero-Information Protocols and Unambiguity in Arthur-Merlin Communication. In Tim Roughgarden, editor, *Proceedings of the 2015 Conference on Innovations in Theoretical Computer Science, ITCS 2015, Rehovot, Israel, January 11-13, 2015*, pages 113–122. ACM, 2015. doi:10.1145/2688073.2688074.
- 24 Mika Göös, Toniann Pitassi, and Thomas Watson. The Landscape of Communication Complexity Classes. *Computational Complexity*, 27(2):245–304, 2018. doi:10.1007/s00037-018-0166-6.
- 25 Yuval Ishai, Eyal Kushilevitz, and Anat Paskin. Secure Multiparty Computation with Minimal Interaction. In Tal Rabin, editor, *Advances in Cryptology - CRYPTO 2010, 30th Annual Cryptology Conference, Santa Barbara, CA, USA, August 15-19, 2010. Proceedings*, volume 6223 of *Lecture Notes in Computer Science*, pages 577–594. Springer, 2010. doi:10.1007/978-3-642-14623-7\_31.
- 26 Yuval Ishai and Hoeteck Wee. Partial Garbling Schemes and Their Applications. In Javier Esparza, Pierre Fraigniaud, Thore Husfeldt, and Elias Koutsoupias, editors, *Automata, Languages, and Programming - 41st International Colloquium, ICALP 2014, Copenhagen, Denmark, July 8-11, 2014, Proceedings, Part I*, volume 8572 of *Lecture Notes in Computer Science*, pages 650–662. Springer, 2014. doi:10.1007/978-3-662-43948-7\_54.
- 27 Ilan Kremer, Noam Nisan, and Dana Ron. On Randomized One-Round Communication Complexity. *Computational Complexity*, 8(1):21–49, 1999. doi:10.1007/s000370050018.
- 28 Eyal Kushilevitz and Noam Nisan. *Communication complexity*. Cambridge University Press, 1997.
- 29 Tianren Liu and Vinod Vaikuntanathan. Breaking the Circuit-Size Barrier in Secret Sharing. To appear in STOC2018, 2018. URL: <https://eprint.iacr.org/2018/333>.
- 30 Tianren Liu, Vinod Vaikuntanathan, and Hoeteck Wee. Conditional Disclosure of Secrets via Non-linear Reconstruction. In Jonathan Katz and Hovav Shacham, editors, *Advances in Cryptology - CRYPTO 2017 - 37th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 20-24, 2017, Proceedings, Part I*, volume 10401 of *Lecture Notes in Computer Science*, pages 758–790. Springer, 2017. doi:10.1007/978-3-319-63688-7\_25.
- 31 Tianren Liu, Vinod Vaikuntanathan, and Hoeteck Wee. Towards Breaking the Exponential Barrier for General Secret Sharing. To appear in Eurocrypt 2018, 2017. URL: <https://eprint.iacr.org/2017/1062>.
- 32 Peter Bro Miltersen, Noam Nisan, Shmuel Safra, and Avi Wigderson. On Data Structures and Asymmetric Communication Complexity. *J. Comput. Syst. Sci.*, 57(1):37–49, 1998. doi:10.1006/jcss.1998.1577.

- 33 Ilan Newman. Private vs. Common Random Bits in Communication Complexity. *Inf. Process. Lett.*, 39(2):67–71, 1991. doi:10.1016/0020-0190(91)90157-D.
- 34 Amit Sahai and Salil P. Vadhan. A complete problem for statistical zero knowledge. *J. ACM*, 50(2):196–249, 2003. doi:10.1145/636865.636868.
- 35 Claude E. Shannon. Communication Theory of Secrecy Systems. *Bell Systems Technical Journal*, 28:656–715, 1949.
- 36 Hung-Min Sun and Shiuh-Pyng Shieh. Secret Sharing in Graph-Based Prohibited Structures. In *Proceedings IEEE INFOCOM '97, The Conference on Computer Communications, Sixteenth Annual Joint Conference of the IEEE Computer and Communications Societies, Driving the Information Revolution, Kobe, Japan, April 7-12, 1997*, pages 718–724. IEEE, 1997. URL: <http://ieeexplore.ieee.org/xpl/mostRecentIssue.jsp?punumber=4979>, doi:10.1109/INFCOM.1997.644525.
- 37 Hoeteck Wee. Dual System Encryption via Predicate Encodings. In Yehuda Lindell, editor, *Theory of Cryptography - 11th Theory of Cryptography Conference, TCC 2014, San Diego, CA, USA, February 24-26, 2014. Proceedings*, volume 8349 of *Lecture Notes in Computer Science*, pages 616–637. Springer, 2014. doi:10.1007/978-3-642-54242-8\_26.
- 38 Hoteck Wee. Personal Communication, 2018.

## A Formal Setup

For a finite set  $A$  we write  $a \stackrel{R}{\leftarrow} A$  to denote a random variable which is sampled uniformly from  $A$ . The *statistical distance* between two discrete random variables,  $X$  and  $Y$ , denoted by  $\Delta(X; Y)$  is defined by  $\Delta(X; Y) := \frac{1}{2} \sum_z |\Pr[X = z] - \Pr[Y = z]|$ . We will also use statistical distance for probability distributions, where for a probability distribution  $D$  the value  $\Pr[D = z]$  is defined to be  $D(z)$ .

► **Definition 13** (CDS). Let  $f : \mathcal{X} \times \mathcal{Y} \rightarrow \{0, 1\}$  be a predicate. Let  $F_A : \mathcal{X} \times \mathcal{Z} \times \mathcal{R} \rightarrow \mathcal{T}_A$  and  $F_B : \mathcal{Y} \times \mathcal{Z} \times \mathcal{R} \rightarrow \mathcal{T}_B$  be deterministic encoding algorithms, where  $\mathcal{Z}$  is the *secret domain*. Then, the pair  $(F_A, F_B)$  is a CDS scheme for  $f$  with *correctness error*  $c$  and *privacy error*  $s$  if the function  $F(x, y, z, r) = (F_A(x, z, r), F_B(y, z, r))$  that corresponds to the joint computation of  $F_A$  and  $F_B$  on a common  $z$  and  $r$ , satisfies the following properties:

1. ( $c$ -Correctness) There exists a deterministic algorithm  $\text{Dec}$ , called a *decoder*, such that for every 1-input  $(x, y)$  of  $f$  and any secret  $z \in \mathcal{Z}$  we have that:

$$\Pr_{r \stackrel{R}{\leftarrow} \mathcal{R}} [\text{Dec}(x, y, F(x, y, z, r)) \neq z] \leq c$$

2. ( $s$ -Privacy) There exists a randomized simulator  $\text{Sim}$  such that for every 0-input  $(x, y)$  of  $f$ , every secret  $z \in \mathcal{Z}$ , and uniformly chosen randomness  $r \stackrel{R}{\leftarrow} \mathcal{R}$  the following holds:

$$\Delta(\text{Sim}(x, y) ; F(x, y, z, r)) \leq s.$$

The *communication complexity* of the CDS protocol is  $(\log |\mathcal{T}_A| + \log |\mathcal{T}_B|)$  and its *randomness complexity* is  $\log |\mathcal{R}|$ . If  $c$  and  $s$  are zeros, such a CDS scheme is called *perfect*.

# Bitcoin: A Natural Oligopoly

Nick Arnosti

Columbia University, New York City, NY, USA  
nicholas.arnosti@gmail.com

S. Matthew Weinberg<sup>1</sup>

Princeton University, Princeton, NJ, USA  
smweinberg@princeton.edu

---

## Abstract

Although Bitcoin was intended to be a decentralized digital currency, in practice, mining power is quite concentrated. This fact is a persistent source of concern for the Bitcoin community.

We provide an explanation using a simple model to capture miners' incentives to invest in equipment. In our model,  $n$  miners compete for a prize of fixed size. Each miner chooses an investment  $q_i$ , incurring cost  $c_i q_i$ , and then receives reward  $\frac{q_i^\alpha}{\sum_j q_j^\alpha}$ , for some  $\alpha \geq 1$ . When  $c_i = c_j$  for all  $i, j$ , and  $\alpha = 1$ , there is a unique equilibrium where all miners invest equally. However, we prove that under seemingly mild deviations from this model, equilibrium outcomes become drastically more centralized. In particular,

- When costs are asymmetric, if miner  $i$  chooses to invest, then miner  $j$  has market share at least  $1 - \frac{c_j}{c_i}$ . That is, if miner  $j$  has costs that are (e.g.) 20% lower than those of miner  $i$ , then miner  $j$  must control at least 20% of the *total* mining power.
- In the presence of economies of scale ( $\alpha > 1$ ), every market participant has a market share of at least  $1 - \frac{1}{\alpha}$ , implying that the market features at most  $\frac{\alpha}{\alpha-1}$  miners in total.

We discuss the implications of our results for the future design of cryptocurrencies. In particular, our work further motivates the study of protocols that minimize “orphaned” blocks, proof-of-stake protocols, and incentive compatible protocols.

**2012 ACM Subject Classification** Theory of computation → Quality of equilibria

**Keywords and phrases** Bitcoin, Cryptocurrencies, Rent-Seeking Competition

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2019.5

**Related Version** Full version available on arXiv at <https://arxiv.org/abs/1811.08572>.

---

<sup>1</sup> Supported by NSF CCF-1717899.





# A Simple Sublinear-Time Algorithm for Counting Arbitrary Subgraphs via Edge Sampling

Sepehr Assadi<sup>1</sup>

Department of Computer and Information Science, University of Pennsylvania,  
Philadelphia, PA, USA  
sassadi@cis.upenn.edu

Michael Kapralov<sup>2</sup>

School of Computer and Communication Sciences, EPFL, Lausanne, Switzerland  
michael.kapralov@epfl.ch

Sanjeev Khanna<sup>3</sup>

Department of Computer and Information Science, University of Pennsylvania,  
Philadelphia, PA, USA  
sanjeev@cis.upenn.edu

---

## Abstract

In the subgraph counting problem, we are given a (large) input graph  $G(V, E)$  and a (small) target graph  $H$  (e.g., a triangle); the goal is to estimate the number of occurrences of  $H$  in  $G$ . Our focus here is on designing *sublinear-time* algorithms for approximately computing number of occurrences of  $H$  in  $G$  in the setting where the algorithm is given *query access* to  $G$ . This problem has been studied in several recent papers which primarily focused on specific families of graphs  $H$  such as triangles, cliques, and stars. However, not much is known about approximate counting of arbitrary graphs  $H$  in the literature. This is in sharp contrast to the closely related subgraph enumeration problem that has received significant attention in the database community as the database join problem. The AGM bound shows that the maximum number of occurrences of any arbitrary subgraph  $H$  in a graph  $G$  with  $m$  edges is  $O(m^{\rho(H)})$ , where  $\rho(H)$  is the *fractional edge-cover* of  $H$ , and enumeration algorithms with matching runtime are known for any  $H$ .

We bridge this gap between subgraph counting and subgraph enumeration by designing a simple sublinear-time algorithm that can estimate the number of occurrences of any arbitrary graph  $H$  in  $G$ , denoted by  $\#H$ , to within a  $(1 \pm \varepsilon)$ -approximation with high probability in  $O(\frac{m^{\rho(H)}}{\#H}) \cdot \text{poly}(\log n, 1/\varepsilon)$  time. Our algorithm is allowed the standard set of queries for general graphs, namely degree queries, pair queries and neighbor queries, plus an additional edge-sample query that returns an edge chosen uniformly at random. The performance of our algorithm matches those of Eden *et al.* [FOCS 2015, STOC 2018] for counting triangles and cliques and extend them to all choices of subgraph  $H$  under the additional assumption of edge-sample queries.

**2012 ACM Subject Classification** Theory of computation  $\rightarrow$  Streaming, sublinear and near linear time algorithms

**Keywords and phrases** Sublinear-time algorithms, Subgraph counting, AGM bound

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2019.6

**Related Version** A full version is available on arXiv [4], <https://arxiv.org/abs/1811.07780>.

**Acknowledgements** We are thankful to the anonymous reviewers of ITCS 2019 for many valuable comments.

---

<sup>1</sup> Supported in part by the National Science Foundation grant CCF-1617851.

<sup>2</sup> Supported in part by ERC Starting Grant 759471.

<sup>3</sup> Supported in part by the National Science Foundation grants CCF-1617851 and CCF-1763514.



## 1 Introduction

Counting (small) subgraphs in massive graphs is a fundamental algorithmic problem, with a wide range of applications in bioinformatics, social network analysis, spam detection and graph databases (see, e.g. [36, 8, 11]). In social network analysis, the ratio of the number of triangles in a network to the number of length 2 paths (known as the clustering coefficient) as well as subgraph counts for larger subgraphs  $H$  have been proposed as important metrics for analyzing massive networks [42]. Similarly, motif counting are popular method for analyzing protein-protein interaction networks in bioinformatics (e.g., [36]). In this paper we consider designing efficient algorithms for this task.

Formally, we consider the following problem: Given a (large) graph  $G(V, E)$  with  $m$  edges and a (small) subgraph  $H(V_H, E_H)$  (e.g., a triangle) and a precision parameter  $\varepsilon \in (0, 1)$ , output a  $(1 \pm \varepsilon)$ -approximation to the number of occurrences of  $H$  in  $G$ . Our goal is to design an algorithm that runs in time *sublinear* in the number  $m$  of edges of  $G$ , and in particular makes a sublinear number of the following types of queries to the graph  $G$ :

- **Degree query  $v$** : the degree  $d_v$  of any vertex  $v \in V$ ;
- **Neighbor query  $(v, i)$** : what vertex is the  $i$ -th neighbor of the vertex  $v \in V$  for  $i \leq d_v$ ;
- **Pair query  $(u, v)$** : test for pair of vertices  $u, v \in V$ , whether or not  $(u, v)$  belongs to  $E$ .
- **Edge-sample query**: sample an edge  $e$  uniformly at random from  $E$ .

The first three queries are the standard baseline queries (see Chapter 10 of Goldreich's book [23]) assumed by nearly all sublinear time algorithms for counting small subgraphs such as triangles or cliques [16, 18] (see [25] for the necessity of using pair queries for counting subgraphs beside stars). The last query is somewhat less standard but is also considered in the literature prior to our work, for example in [2] for counting stars in sublinear time, and in [19] in the context of lower bounds for subgraph counting problems.

### 1.1 Our Contributions

For the sake of clarity, we suppress any dependencies on the approximation parameter  $\varepsilon$ ,  $\log n$ -terms, and the size of graph  $H$ , using the  $O^*(\cdot)$  notation. Our results are parameterized by the *fractional edge-cover number* of the subgraph  $H$  (see Section 3 for the formal definition). Our goal in this paper is to approximately compute the number of occurrences  $\#H$  of  $H$  in  $G$ , formally defined as number of subgraphs  $H'$  of  $G$  such that  $H$  and  $H'$  are isomorphic.

► **Theorem 1.** *There exists a randomized algorithm that given  $\varepsilon \in (0, 1)$ , a subgraph  $H$ , and a query access to the input graph  $G$ , with high probability outputs a  $(1 \pm \varepsilon)$  approximation to the number of occurrences of  $H$  in  $G$ , denoted by  $\#H$ , using:*

$$O^*\left(\min\left\{m, \frac{m^{\rho(H)}}{\#H}\right\}\right) \text{ queries and } O^*\left(\frac{m^{\rho(H)}}{\#H}\right) \text{ time.}$$

*The algorithm uses degree, neighbor, pair, and edge-sample queries.*

Since the fractional edge-cover number of any  $k$ -clique  $K_k$  is  $k/2$ , as a corollary of Theorem 1, we obtain sublinear algorithms for counting triangles, and in general  $k$ -cliques using

$$O^*\left(\min\left\{m, \frac{m\sqrt{m}}{\#K_3}\right\}\right) \text{ and } O^*\left(\min\left\{m, \frac{m^{k/2}}{\#K_k}\right\}\right),$$

queries respectively. These bounds match the previous results of Eden *et al.* [16, 18] modulo an additive term of  $O^*\left(\frac{n}{(\#K_3)^{1/3}}\right)$  for triangles in [16] and  $O^*\left(\frac{n}{(\#K_k)^{1/k}}\right)$  for arbitrary cliques



in [18] which is needed in the absence of edge-sample queries that are not used by [16, 18]. Our bounds settle a conjecture of Eden and Rosenbaum [19] in the affirmative by showing that one can avoid the aforementioned additive terms depending on  $n$  in query complexity by allowing edge-sample queries. We now elaborate more on different aspects of Theorem 1.

**AGM Bound and Database Joins.** The problem of *enumerating* all occurrences of a graph  $H$  in a graph  $G$  and, more generally, the database join problem, has been considered extensively in the literature. A fundamental question here is that given a database with  $m$  entries (e.g. a graph  $G$  with  $m$  edges) and a conjunctive query  $H$  (e.g. a small graph  $H$ ), what is the maximum possible size of the output of the query (e.g., number of occurrences of  $H$  in  $G$ )? The AGM bound of Atserias, Grohe and Marx [5] provides a tight upper bound of  $m^{\rho(H)}$  (up to constant factors), where  $\rho(H)$  is the fractional edge cover of  $H$ . The AGM bound applies to databases with several relations, and the fractional edge cover in question is weighted according to the sizes of the different relations. A similar bound on the number of occurrences of a hypergraph  $H$  inside a hypergraph  $G$  with  $m$  hyperedges was proved earlier by Friedgut and Kahn [22], and the bound for graphs is due to Alon [3]. Recent work of Ngo et al. [37] gave algorithms for evaluating database joins in time bounded by worst case output size for a database with the same number of entries. When instantiated for the subgraph enumeration problem, their result gives an algorithm for enumerating all occurrences of  $H$  in a graph  $G$  with  $m$  edges in time  $O(m^{\rho(H)})$ .

Our Theorem 1 is directly motivated by the connection between subgraph counting and subgraph enumeration problems and the AGM bound. In particular, Theorem 1 provides a natural analogue of AGM bound for estimation algorithms by stating that if the number of occurrences  $H$  is  $\#H \leq m^{\rho(H)}$ , then a  $(1 \pm \varepsilon)$ -approximation to  $\#H$  can be obtained in  $O^*(\frac{m^{\rho(H)}}{\#H})$  time. Additionally, as we show in Section 4.3, Theorem 1 can be easily extended to the more general problem of database join size estimation (for binary relations). This problem corresponds to a subgraph counting problem in which the graphs  $G$  and  $H$  are both *edge-colored* and our goal is to count the number of copies of  $H$  in  $G$  with the same colors on edges. Our algorithm can solve this problem also in  $O^*(\frac{m^{\rho(H)}}{\#H_c})$  time where  $\#H_c$  denotes the number of copies of  $H$  with the same colors in  $G$ .

**Optimality of Our Bounds.** Our algorithm in Theorem 1 is optimal from different points of view. Firstly, by a lower bound of [19] (building on [16, 18]), the bounds achieved by our algorithm when  $H$  is any  $k$ -clique is optimal among all algorithms with the same query access (including the edge-sample query). In Theorem 15, we further prove a lower bound showing that for *odd cycles* as well, the bounds achieved by Theorem 1 are optimal. These results hence suggest that Theorem 1 is *existentially optimal*: there exists several natural choices for  $H$  such that Theorem 1 achieves the optimal bounds. However, there also exist choices of  $H$  for which the bounds in Theorem 1 are suboptimal. In particular, Aliakbarpour *et al.* [2] presented an algorithm for estimating occurrences of any star  $S_\ell$  for  $\ell \geq 1$  using  $O^*(\frac{m}{(\#S_\ell)^{1/\ell}})$  queries in our query model (including edge-sample queries) which is always at least as good as our bound in Theorem 1, but potentially can be better. On the other hand, in the full version of the paper [4], we show that our current algorithm, with almost no further modification, in fact achieves this stronger bound *using a different analysis*.

Additionally, as we pointed out before, our algorithm can solve the more general database join size estimation for binary relations, or equivalently the subgraph counting problem with colors on edges. In Theorem 16, we prove that for this more general problem, our algorithm in Theorem 1 indeed achieves optimal bounds for *all choices* of the subgraph  $H$ .

**Edge-Sample Queries.** The edge-sample query that we assume is not part of the standard access model for sublinear algorithms, namely the “general graph” query model (see, e.g. [32]). Nonetheless, we find allowing for this query “natural” owing to the following factors:

*Theoretical implementation.* Edge sampling queries can be implemented with an  $\tilde{O}(n/\sqrt{m})$  multiplicative overhead in query and time using the recent result of [20], or with an  $O(n)$  additive preprocessing time (which is still sublinear in  $m$ ) by querying degrees of all vertices. Hence, we can focus on designing algorithms by allowing these queries and later replacing them by either of the above implementations in a black-box way at a certain additional cost.

*Practical implementation.* Edge sampling is a common practice in analyzing social networks [34, 33] or biological networks [1]. Another scenario when random edge sampling is possible is when we can access a random location of the memory that is used to store the graph. To quote [2]: “because edges normally take most of the space for storing graphs, an access to a random memory location where the adjacency list is stored, would readily give a random edge.” Hence, assuming edge sampling queries can be considered valid in many scenarios.

*Understanding the power of random edge queries.* Edge sampling is a critical component of various sublinear time algorithms for graph estimation [16, 17, 2, 18, 20]. However, except for [2] that also assumed edge-sample queries, all these other algorithms employ different workarounds to these queries. As we show in this paper, decoupling these workarounds from the rest of the algorithm by allowing edge-sample queries results in considerably simpler and more general algorithms for subgraph counting and is hence worth studying on its own. We also mention that studying the power of edge-sample queries has been cast as an open question in [19] as well.

**Applications to Streaming Algorithms.** Subgraph counting is also one of the most studied problems in the graph streaming model (see, e.g. [6, 28, 10, 31, 9, 27, 41, 35, 13, 7] and references therein). In this model, the edges of the input graph are presented one by one in a stream; the algorithm makes a single or a small number of passes over the stream and outputs the answer after the last pass. The goal here is to minimize the memory used by the algorithm (similar-in-spirit to minimizing the query complexity in the query model).

Our algorithm in Theorem 1 can be directly adapted to the streaming model, resulting in an algorithm for subgraph counting that makes  $O(1)$  passes over the stream and uses a memory of size  $O^*\left(\min\left\{m, \frac{m^{\rho(H)}}{\#H}\right\}\right)$ . For the case of counting triangles and cliques, the space complexity of our algorithm matches the best known algorithms of McGregor *et al.* [35] and Bera and Chakrabarti [7] which are known to be optimal [7]. To the best of our knowledge, the only previous streaming algorithms for counting arbitrary subgraphs  $H$  are those of Kane *et al.* [31] and Bera and Chakrabarti [7] that use, respectively, one pass and  $O^*\left(\frac{m^{2 \cdot |E(H)|}}{(\#H)^2}\right)$  space, and two passes and  $O^*\left(\frac{m^{\beta(H)}}{\#H}\right)$  space, where  $\beta(H)$  is the *integral* edge-cover number of  $H$ . As  $\rho(H) \leq \beta(H) \leq |E(H)|$  by definition and  $\#H \leq m^{\rho(H)}$  by AGM bound, the space complexity of our algorithm is always at least as good as the ones in [31, 7] but potentially can be much smaller.

## 1.2 Main Ideas in Our Algorithm

Our starting point is the AGM bound which implies that the number of “potential copies” of  $H$  in  $G$  is at most  $m^{\rho(H)}$ . Our goal of estimating  $\#H$  then translates to counting how many of these potential copies form an actual copy of  $H$  in  $G$ . A standard approach at this point is the *Monte Carlo method*: sample a potential copy of  $H$  in  $G$  *uniformly at random*

and check whether it forms an actual copy of  $H$  or not; a simple exercise in concentration inequalities then implies that we only need  $O(\frac{m^{\rho(H)}}{\#H})$  many *independent* samples to get a good estimate of  $\#H$ .

This approach however immediately runs into a technical difficulty. Given only a query access to  $G$ , it is not at all clear how to sample a potential copy of  $H$  from the list of all potential copies. Our first task is then to design a procedure for sampling potential copies of  $H$  from  $G$ . In order to do so, we again consider the AGM bound and the optimal fractional edge-cover that is used to derive this bound. We first prove a simple structural result that states that an optimal fractional edge-cover of  $H$  can be supported only on edges that form a disjoint union of *odd cycles* and *stars* (in  $H$ ). This allows us to decompose  $H$  into a collection of odd cycles and stars and treat any arbitrary subgraph  $H$  as a collection of these simpler subgraphs that are suitably connected together.

The above decomposition reduces the task of sampling a potential copy of  $H$  to sampling a collection of odd cycles and stars. Sampling an odd cycle  $C_{2k+1}$  on  $2k+1$  edges is as follows: sample  $k$  edges  $e_1, \dots, e_k$  uniformly at random from  $G$ ; pick one of the endpoints of  $e_1$  and sample a vertex  $v$  from the neighborhood of this endpoint uniformly at random. With some additional care, one can show that the tuple  $(e_1, \dots, e_k, v)$  sampled here is enough to identify an odd cycle of length  $2k+1$  uniquely. To sample a star  $C_\ell$  with  $\ell$  petals, we sample a vertex  $v$  from  $G$  with probability proportional to its degree (by sampling a random edge and picking one of the two endpoints uniformly), and then sample  $\ell$  vertices  $w_1, \dots, w_\ell$  from the neighborhood of  $v$ . Again, with some care, this allows us to sample a potential copy of a star  $S_\ell$ . We remark that these sampling procedures are related to sampling triangles in [16] and stars in [2]. Finally, to sample a potential copy of  $H$ , we simply sample all its odd cycles and stars in the decomposition using the method above. We should note right away that this however does *not* result in a uniformly at random sample of potential copies of  $H$  as various parameters of the graph  $G$ , in particular degrees of vertices, alter the probability of sampling each potential copy.

The next and paramount step is then how to use the samples above to estimate the value of  $\#H$ . Obtaining an unbiased estimator of  $\#H$  based on these samples is not hard as we can identify the probability each potential copy is sampled with in this process (which is a function of degrees of vertices of the potential copy in  $G$ ) and reweigh each sample accordingly. Nevertheless, the variance of a vanilla variant of this sampling and reweighing approach is quite large for our purpose. To fix this, we use an idea similar to that of [16] for counting triangles: sample a “partial” potential copy of  $H$  first and fix it; sample *multiple* “extensions” of this partial potential copy to a complete potential copy and use the average of estimates based on each extension to reduce the variance. More concretely, this translates to sampling multiple copies of the first cycle for the decomposition and for each sampled cycle, recursively sampling multiple copies of the remainder of  $H$  as specified by the decomposition. A careful analysis of this recursive process – which is the main technical part of the paper – allows us to bound the variance of the estimator by  $O(m^{\rho(H)} \cdot (\#H))$ . Repeating such an estimator  $O(\frac{m^{\rho(H)}}{\#H})$  times independently and taking the average value then gives us a  $(1 \pm \varepsilon)$ -approximation to  $\#H$  by a simple application of Chebyshev’s inequality.

### 1.3 Further Related Work

In addition to the previous work in [16, 18, 2] that are already discussed above, sublinear-time algorithms for estimating subgraph counts and related parameters such as average degree and degree distribution moments have also been studied in [21, 24, 25, 17]. Similarly, sublinear-

time algorithms are also studied for estimating other graph parameters such as weight of the minimum spanning tree [15, 12, 14] or size of a maximum matching or a minimum vertex cover [40, 38, 43, 26, 39] (this is by no means a comprehensive summary of previous results).

Subgraph counting has also been studied extensively in the graph streaming model (see, e.g. [6, 28, 10, 31, 9, 27, 41, 35, 13, 7, 30, 29] and references therein). In this model, the edges of the input graph are presented one by one in a stream; the algorithm makes a single or a small number of passes over the stream and outputs the answer after the last pass. The goal in this model is to minimize the memory used by the algorithm similar-in-spirit to minimizing the query complexity in our query model. However, the streaming algorithms typically require reading the entire graph in the stream which is different from our goal in sublinear-time algorithms.

## 2 Preliminaries

**Notation.** For any integer  $t \geq 1$ , we let  $[t] := \{1, \dots, t\}$ . For any event  $\mathcal{E}$ ,  $\mathbb{I}(\mathcal{E}) \in \{0, 1\}$  is an indicator denoting whether  $\mathcal{E}$  happened or not. For a graph  $G(V, E)$ ,  $V(G) := V$  denotes the vertices and  $E(G) := E$  denotes the edges. For a vertex  $v \in V$ ,  $N(v)$  denotes the neighbors of  $v$ , and  $d_v := |N(v)|$  denotes the degree of  $v$ .

To any edge  $e = \{u, v\}$  in  $G$ , we assign two directed edges  $\vec{e}_1 = (u, v)$  and  $\vec{e}_2 = (v, u)$  called the directed copies of  $e$  and let  $\vec{E}$  be the set of all these directed edges. We also fix a total ordering  $\prec$  on vertices whereby for any two vertices  $u, v \in V$ ,  $u \prec v$  iff  $d_u < d_v$ , or  $d_u = d_v$  and  $u$  appears before  $v$  in the lexicographic order. To avoid confusion, we use letters  $a, b$  and  $c$  to denote the vertices in the subgraph  $H$ , and letters  $u, v$  and  $w$  to denote the vertices of  $G$ .

We use the following standard variant of Chebyshev's inequality.

► **Proposition 2.** For any random variable  $X$  and integer  $t \geq 1$ ,  $\Pr(|X - \mathbb{E}[X]| \geq t) \leq \frac{\text{Var}[X]}{t^2}$ .

We also recall the law of total variance that states the for two random variables  $X$  and  $Y$ ,

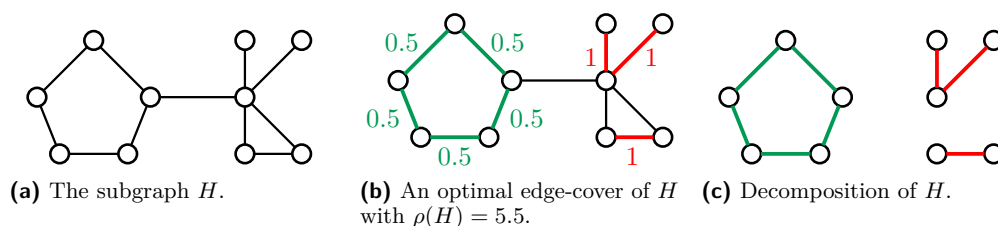
$$\text{Var}[Y] = \mathbb{E}_x(\text{Var}[Y | X = x]) + \text{Var}_x[\mathbb{E}[Y | X = x]]. \quad (1)$$

**Assumption on Size of Subgraph  $H$ .** Throughout the paper, we assume that the size of the subgraph  $H$  is a fixed constant independent of the size of the graph  $G$  and hence we suppress the dependency on size of  $H$  in various bounds in our analysis using  $O$ -notation.

## 3 A Graph Decomposition Using Fractional Edge-Covers

In this section, we give a simple decomposition of the subgraph  $H$  using fractional edge-covers. We start by defining fractional edge-covers formally (see also Figure 1).

► **Definition 3 (Fractional Edge-Cover Number).** A fractional edge-cover of  $H(V_H, E_H)$  is a mapping  $\psi : E_H \rightarrow [0, 1]$  such that for each vertex  $a \in V_H$ ,  $\sum_{e \in E_H, a \in e} \psi(e) \geq 1$ . The fractional edge-cover number  $\rho(H)$  of  $H$  is the minimum value of  $\sum_{e \in E_H} \psi(e)$  among all fractional edge-covers  $\psi$ .



■ **Figure 1** Illustration of the our decomposition for  $H$  based on fractional edge-covers.

The fractional edge-cover number of a graph can be computed by the following LP:

$$\begin{aligned} \rho(H) &= \text{minimize} && \sum_{e \in E(H)} x_e \\ &\text{subject to} && \sum_{e \in E_H: a \in e} x_e \geq 1 \text{ for all vertices } a \in V(H). \end{aligned} \quad (2)$$

The following lemma is the key to our decomposition. The proof is based on standard ideas in linear programming and is postponed to the full version of the paper [4].

► **Lemma 4.** *Any subgraph  $H$  admits an optimal fractional edge-cover  $x^*$  such that the support of  $x^*$ , denoted by  $\text{SUPP}(x^*)$ , is a collection of vertex-disjoint odd cycles and star graphs, and,*

1. *for every odd cycle  $C \in \text{SUPP}(x^*)$ ,  $x_e^* = 1/2$  for all  $e \in C$ ;*
2. *for every edge  $e \in \text{SUPP}(x^*)$  that does not belong to any odd cycle,  $x_e = 1$ .*

### 3.1 The Decomposition

We now present the decomposition of  $H$  using Lemma 4. Let  $x^*$  be an optimal fractional edge-cover in Lemma 4 and let  $\mathcal{C}_1, \dots, \mathcal{C}_o$  be the odd-cycles in the support of  $x^*$  and  $\mathcal{S}_1, \dots, \mathcal{S}_s$  be the stars. We define  $\mathcal{D}(H) := \{\mathcal{C}_1, \dots, \mathcal{C}_o, \mathcal{S}_1, \dots, \mathcal{S}_s\}$  as the decomposition of  $H$  (see Figure 1 for an illustration).

For every  $i \in [o]$ , let the length of the odd cycle  $\mathcal{C}_i$  be  $2k_i + 1$  (i.e.,  $\mathcal{C}_i = C_{2k_i+1}$ ); we define  $\rho_i^C := k_i + 1/2$ . Similarly, for every  $j \in [s]$ , let the number of petals in  $\mathcal{S}_j$  be  $\ell_j$  (i.e.,  $\mathcal{S}_j = S_{\ell_j}$ ); we define  $\rho_j^S := \ell_j$ . By Lemma 4,

$$\rho(H) = \sum_{i=1}^o \rho_i^C + \sum_{j=1}^s \rho_j^S. \quad (3)$$

Recall that by AGM bound, the total number of copies of  $H$  possible in  $G$  is  $m^{\rho(H)}$ . We also use the following simple lemma which is a direct corollary of the AGM bound.

► **Lemma 5.** *Let  $I := \{i_1, \dots, i_o\}$  and  $J := \{j_1, \dots, j_s\}$  be subsets of  $[o]$  and  $[s]$ , respectively. Suppose  $\tilde{H}$  is the subgraph of  $H$  on vertices of the odd cycles  $\mathcal{C}_{i_1}, \dots, \mathcal{C}_{i_o}$  and stars  $\mathcal{S}_{j_1}, \dots, \mathcal{S}_{j_s}$ . Then the total number of copies of  $\tilde{H}$  in  $G$  is at most  $m^{\rho(\tilde{H})}$  for  $\rho(\tilde{H}) \leq \sum_{i \in I} \rho_i^C + \sum_{j \in J} \rho_j^S$ .*

**Proof.** Let  $x^*$  denote the optimal value of LP (2) in  $\mathcal{D}(H)$ . Define  $y^*$  as the projection of  $x^*$  to edges present in  $\tilde{H}$ . It is easy to see that  $y^*$  is a feasible solution for LP (2) of  $\tilde{H}$  with value  $\sum_{i \in I} \rho_i^C + \sum_{j \in J} \rho_j^S$ . The lemma now follows from AGM bound for  $\tilde{H}$ . ◀

### 3.2 Profiles of Cycles, Stars, and Subgraphs

We conclude this section by specifying the representation of the potential occurrences of the subgraph  $H$  in  $G$  based on the decomposition  $\mathcal{D}(H)$ .

*Odd cycles:* We represent a potential occurrence of an odd cycle  $C_{2k+1}$  in  $G$  as follows. Let  $\mathbf{e} = (\vec{e}_1, \dots, \vec{e}_k) \in \vec{E}^k$  be an ordered tuple of  $k$  directed copies of edges in  $G$  and suppose  $\vec{e}_i := (u_i, v_i)$  for all  $i \in [k]$ . Define  $u_{\mathbf{e}}^* = u_1$  and let  $w$  be any vertex in  $N(u_{\mathbf{e}}^*)$ . We refer to any such collection  $(\mathbf{e}, w)$  as a *profile* of  $C_{2k+1}$  in  $G$ . We say that “the profile  $(\mathbf{e}, w)$  forms a cycle  $C_{2k+1}$  in  $G$ ” iff (i)  $u_1$  is the smallest vertex on the cycle according to  $\prec$ , (ii)  $v_1 \prec w$ , and (iii) the edges  $(u_1, v_1), (v_1, u_2), \dots, (u_k, v_k), (v_k, w), (w, u_1)$  all exist in  $G$  and hence there is a copy of  $C_{2k+1}$  on vertices  $\{u_1, v_1, u_2, v_2, \dots, u_k, v_k, w\}$  in  $G$ . Note that under this definition and our definition of  $\#C_{2k+1}$ , each copy of  $C_{2k+1}$  correspond to exactly one profile  $(\mathbf{e}, w)$  and vice versa. As such,

$$\#C_{2k+1} = \sum_{\mathbf{e} \in \vec{E}^k} \sum_{w \in N(u_{\mathbf{e}}^*)} \mathbb{I}\left((\mathbf{e}, w) \text{ forms a cycle } C_{2k+1} \text{ in } G\right). \quad (4)$$

*Stars:* We represent a potential occurrence of a star  $S_\ell$  in  $G$  by  $(v, \mathbf{w})$  where  $v$  is the center of the star and  $\mathbf{w} = (w_1, \dots, w_\ell)$  are the  $\ell$  petals. We refer to  $(v, \mathbf{w})$  as a *profile* of  $S_\ell$  in  $G$ . We say that “the profile  $(v, \mathbf{w})$  forms a star  $S_\ell$  in  $G$ ” iff (i)  $|\mathbf{w}| > 1$ , or (ii)  $(\ell = 1)$   $|\mathbf{w}| = 1$  and  $v \prec w_1$ ; in both cases there is a copy of  $S_\ell$  on vertices  $v, w_1, \dots, w_\ell$ . Under this definition, each copy of  $S_\ell$  corresponds to exactly one profile  $(v, \mathbf{w})$ . As such,

$$\#S_\ell = \sum_{v \in V} \sum_{\mathbf{w} \in N(v)^\ell} \mathbb{I}\left((v, \mathbf{w}) \text{ forms a star } S_\ell \text{ in } G\right). \quad (5)$$

*Arbitrary subgraphs:* We represent a potential occurrence of  $H$  in  $G$  by an  $(o + s)$ -tuple  $\mathbf{R} := ((\mathbf{e}_1, \mathbf{w}_1), \dots, (\mathbf{e}_o, \mathbf{w}_o), (v_1, \mathbf{w}_1), \dots, (v_s, \mathbf{w}_s))$  where  $(\mathbf{e}_i, \mathbf{w}_i)$  is a profile of the cycle  $C_i$  in  $\mathcal{D}(H)$  and  $(v_j, \mathbf{w}_j)$  is a profile of the star  $S_j$ . We refer to  $\mathbf{R}$  as a *profile* of  $H$  and say that “the profile  $\mathbf{R}$  forms a copy of  $H$  in  $G$ ” iff (i) each profile forms a corresponding copy of  $C_i$  or  $S_j$  in  $\mathcal{D}(H)$ , and (ii) the remaining edges of  $H$  between vertices specified by  $\mathbf{R}$  all are present in  $G$  (note that by definition of the decomposition  $\mathcal{D}(H)$ , all vertices of  $H$  are specified by  $\mathbf{R}$ ). As such,

$$\#H = \sum_{\mathbf{R}} \mathbb{I}\left(\mathbf{R} \text{ forms a copy of } H \text{ in } G\right) \cdot f(H), \quad (6)$$

for a fixed constant  $f(H)$  depending only on  $H$  as defined below. Let  $\pi : V_H \rightarrow V_H$  be an automorphism of  $H$ . Let  $C_1, \dots, C_o, S_1, \dots, S_s$  denote the cycles and stars in the decomposition of  $H$ . We say that  $\pi$  is decomposition preserving if for every  $i = 1, \dots, o$  cycle  $C_i$  is mapped to a cycle of the same length and for every  $i = 1, \dots, s$  star  $S_i$  is mapped to a star with the same number of petals. Let the number of decomposition preserving automorphisms of  $H$  be denoted by  $Z$ , and define  $f(H) = 1/Z$ . Define the quantity  $\widetilde{\#H} := \sum_{\mathbf{R}} \mathbb{I}\left(\mathbf{R} \text{ forms a copy of } H \text{ in } G\right)$  which is equal to  $\#H$  modulo the scaling factor of  $f(H)$ . It is immediate that estimating  $\#H$  and  $\widetilde{\#H}$  are equivalent to each other and hence in the rest of the paper, with a slight abuse of notation, we use  $\#H$  and  $\widetilde{\#H}$  interchangeably.

## 4 A Sublinear-Time Algorithm for Subgraph Counting

We now present our sublinear time algorithm for approximately counting number of any given arbitrary subgraph  $H$  in an underlying graph  $G$  and prove Theorem 1. The main component of our algorithm is an unbiased estimator random variable for  $\#H$  with low

variance. The algorithm in Theorem 1 is then obtained by simply repeating this unbiased estimator in parallel enough number of times (based on the variance) and outputting the average value of these estimators.

#### 4.1 A Low-variance Unbiased Estimator for $\#H$

We present a low-variance unbiased estimator for  $\#H$  in this section. Our algorithm is a sampling based algorithm. In the following, we first introduce two separate subroutines for sampling odd cycles (**odd-cycle-sampler**) and stars (**star-sampler**), and then use these components in conjunction with the decomposition we introduced in Section 3, to present our full algorithm. We should right away clarify that **odd-cycle-sampler** and **star-sampler** are not exactly sampling a cycle or a star, but rather sampling a set of vertices and edges (in a non-uniform way) that can potentially form a cycle or star in  $G$ , i.e., they sample a profile of these subgraphs defined in Section 3.2.

##### The odd-cycle-sampler Algorithm

We start with the following algorithm for sampling an odd cycle  $C_{2k+1}$  for some  $k \geq 1$ . This algorithm outputs a simple data structure, named the *cycle-sampler tree*, that provides a convenient representation of the samples taken by our algorithm (see Definition 6 immediately after the description of the algorithm). This data structure can be easily avoided when designing a cycle counting algorithm, but will be quite useful for reasoning about the recursive structure of our sampling algorithm for general graphs  $H$ .

---

**Algorithm 1** `odd-cycle-sampler`( $G, C_{2k+1}$ ).

---

1. Sample  $k$  directed edges  $e := (\vec{e}_1, \dots, \vec{e}_k)$  uniformly at random (with replacement) from  $G$  with the constraint that for  $\vec{e}_1 = (u_1, v_1)$ ,  $u_1 \prec v_1$ .
  2. Let  $u_e^* := u_1$  and let  $d_e^* := d_{u_e^*}$ .
  3. For  $i = 1$  to  $t_e := \lceil d_e^* / \sqrt{m} \rceil$ : Sample a vertex  $w_i$  uniformly at random from  $N(u_e^*)$ .
  4. Let  $w := (w_1, \dots, w_{t_e})$ . Return the cycle-sampler tree  $\mathcal{T}(e, w)$  (see Definition 6).
- 

► **Definition 6** (Cycle-Sampler Tree). The cycle-sampler tree  $\mathcal{T}(e, w)$  for the tuple  $(e, w)$  sampled by `odd-cycle-sampler`( $G, C_{2k+1}$ ) is the following 2-level tree  $\mathcal{T}$ :

- Each node  $\alpha$  of the tree contains two attributes:  $\text{label}[\alpha]$  which consists of some of the edges and vertices in  $(e, w)$ , and an integer  $\text{value}[\alpha]$ .
- For the root  $\alpha_r$  of  $\mathcal{T}$ ,  $\text{label}[\alpha_r] := e$  and  $\text{value}[\alpha_r] := (2m)^k / 2$ . ( $\text{value}[\alpha_r]$  is equal to the inverse of the probability that  $e$  is sampled by `odd-cycle-sampler`).
- The root  $\alpha_r$  has  $t_e$  child-nodes in  $\mathcal{T}$  for a parameter  $t_e = \lceil d_e^* / \sqrt{m} \rceil$  (consistent with line 3 of `odd-cycle-sampler`( $G, C_{2k+1}$ ) above).
- For the  $i$ -th child-node  $\alpha_i$  of root,  $i \in [t_e]$ ,  $\text{label}[\alpha_i] := w_i$  and  $\text{value}[\alpha_i] := d_e^*$  ( $\text{value}[\alpha_i]$  is equal to the inverse of the probability that  $w_i$  is sampled by `odd-cycle-sampler`, conditioned on  $e$  being sampled).

Moreover, for each root-to-leaf path  $\mathcal{P}_i := (\alpha_r, \alpha_i)$  (for  $i \in [t_e]$ ), define  $\text{label}[\mathcal{P}_i] := \text{label}[\alpha_r] \cup \text{label}[\alpha_i]$  and  $\text{value}[\mathcal{P}_i] := \text{value}[\alpha_r] \cdot \text{value}[\alpha_i]$  ( $\text{label}[\mathcal{P}_i]$  is a profile of the cycle  $C_{2k+1}$  as defined in Section 3.2).



`odd-cycle-sampler` can be implemented in our query model by using  $k$  edge-sample queries (and picking the correct direction for  $e_1$  based on  $\prec$  and one of the two directions uniformly at random for the other edges) in Line (1), two degree queries in Line (2), and one neighbor query in Line (3). This results in  $O(k)$  queries in total for one iteration of the for-loop in Line (3). As such, the total query complexity of `odd-cycle-sampler` is  $O(t_e)$  (recall that  $k$  is a constant). It is also straightforward to verify that we can compute the cycle-sampler tree  $\mathcal{T}$  of an execution of `odd-cycle-sampler` with no further queries and in  $O(t_e)$  time. We bound the query complexity of this algorithm by bounding the expected number of iterations in the for-loop. The proof is postponed to the full version [4].

► **Lemma 7.** *For the parameter  $t_e$  in Line (3) of `odd-cycle-sampler`,  $\mathbb{E}[t_e] = O(1)$ .*

We now define a process for estimating the number of odd cycles in a graph using the information stored in the cycle-sampler tree and the `odd-cycle-sampler` algorithm. While we do not use this process in a black-box way in our main algorithm, abstracting it out makes the analysis of our main algorithm simpler to follow and more transparent, and serves as a warm-up for our main algorithm.

**Warm-up: An Estimator for Odd Cycles.** Let  $\mathcal{T} := \text{odd-cycle-sampler}(G, C_{2k+1})$  be the output of an invocation of `odd-cycle-sampler`. Note that the cycle-sampler tree  $\mathcal{T}$  is a random variable depending on the randomness of `odd-cycle-sampler`. We define the random variable  $X_i$  such that  $X_i := \text{label}[\mathcal{P}_i]$  for the  $i$ -th root-to-leaf path iff  $\text{label}[\mathcal{P}_i]$  forms a copy of  $C_{2k+1}$  in  $G$  and otherwise  $X_i := 0$  (according to the definition of Section 3). We further define  $Y := \frac{1}{t_e} \cdot \sum_{i=1}^{t_e} X_i$  (note that  $t_e$  is also a random variable). Our estimator algorithm can compute the value of these random variables using the information stored in the tree  $\mathcal{T}$  plus additional  $O(k) = O(1)$  queries for each of the  $t_e$  root-to-leaf path  $\mathcal{P}_i$  to detect whether  $(e, w_i)$  forms a copy of  $H$  or not. Thus, the query complexity and runtime of the estimator is still  $O(t_e)$  (which in expectation is  $O(1)$  by Lemma 7). The expectation and variance of the estimator can be bounded as follows (the proof is in the full version [4]).

► **Lemma 8.** *For the random variable  $Y$  associated with `odd-cycle-sampler`( $G, C_{2k+1}$ ),*

$$\mathbb{E}[Y] = (\#C_{2k+1}), \quad \text{Var}[Y] \leq (2m)^k \sqrt{m} \cdot \mathbb{E}[Y].$$

## The star-sampler Algorithm

We now give an algorithm for sampling a star  $S_\ell$  with  $\ell$  petals. Similar to `odd-cycle-sampler`, this algorithm also outputs a simple data structure, named the *star-sampler tree*, that provides a convenient representation of the samples taken by our algorithm (see Definition 9, immediately after the description of the algorithm). This data structure can be easily avoided when designing a star counting algorithm, but will be quite useful for reasoning about the recursive structure of our sampling algorithm for general graphs  $H$ .

► **Definition 9 (Star-Sampler Tree).** The star-sampler tree  $\mathcal{T}(v, \mathbf{w})$  for the tuple  $(v, \mathbf{w})$  sampled by `star-sampler`( $G, S_\ell$ ) is the following 2-level tree  $\mathcal{T}$  (with the same attributes as in Definition 6) with only two nodes:

- For the root  $\alpha_r$  of  $\mathcal{T}$ ,  $\text{label}[\alpha_r] := v$  and  $\text{value}[\alpha_r] := 2m/d_v$ .  
( $\text{value}[\alpha_r]$  is equal to the inverse of the probability that  $v$  is sampled by `star-sampler`).
- The root  $\alpha_r$  has exactly one child-node  $\alpha_l$  in  $\mathcal{T}$  with  $\text{label}[\alpha_l] = \mathbf{w} = (w_1, \dots, w_\ell)$  and  $\text{value}[\alpha_l] = \binom{d_v}{\ell}$ .  
( $\text{value}[\alpha_l]$  is equal to the inverse of the probability that  $\mathbf{w}$  is sampled by `star-sampler`, conditioned on  $v$  being sampled).



---

**Algorithm 2** `star-sampler`( $G, S_\ell$ ).

---

1. Sample a vertex  $v \in V$  chosen with probability proportional to its degree in  $G$  (i.e., for any vertex  $u \in V$ ,  $\Pr(u \text{ is chosen as the vertex } v) = d_u/2m$ ).
  2. Sample  $\ell$  vertices  $\mathbf{w} := (w_1, \dots, w_\ell)$  from  $N(v)$  uniformly at random (without replacement).
  3. Return the star-sampler tree  $\mathcal{T}(v, \mathbf{w})$  (see Definition 9).
- 

Moreover, for the root-to-leaf path  $\mathcal{P} := (\alpha_r, \alpha_\ell)$ , we define  $\text{label}[\mathcal{P}] := \text{label}[\alpha_r] \cup \text{label}[\alpha_\ell]$  and  $\text{value}[\mathcal{P}] := \text{value}[\alpha_r] \cdot \text{value}[\alpha_\ell]$ . ( $\text{label}[\mathcal{P}]$  is a representation of the star  $S_\ell$  as defined in Section 3.2).

`star-sampler` can be implemented in our query model by using one edge-sample query in Line (1) and then picking one of the endpoints uniformly at random, a degree query to determine the degree of  $v$ , and  $\ell$  neighbor queries in Line (2), resulting in  $O(\ell)$  queries in total. It is also straightforward to verify that we can compute the star-sampler tree  $\mathcal{T}$  of an execution of `star-sampler` with no further queries and in  $O(1)$  time.

We again define a process for estimating the number of stars in a graph using the information stored in the star-sampler tree and the `star-sampler` algorithm, as a warm-up to our main result in the next section.

**Warm-up: An Estimator for Stars.** The star-sampler tree  $\mathcal{T}$  is a random variable depending on the randomness of `star-sampler`. We define the random variable  $X$  such that  $X := \text{value}[\mathcal{P}]$  for the root-to-leaf path of  $\mathcal{T}$  iff  $\text{label}[\mathcal{P}]$  forms a copy of  $S_\ell$  in  $G$  and otherwise  $X := 0$ . Our estimator algorithm can compute the value of this random variable using only the information stored in the tree  $\mathcal{T}$  with no further queries to the graph (by simply checking if all  $w_i$ 's in  $\mathbf{w}$  are distinct). As such, the query complexity and runtime of the estimator algorithm is still  $O(1)$ . The proof of the following lemma is postponed to the full version [4].

► **Lemma 10.** *For the random variable  $X$  associated with `star-sampler`( $G, S_\ell$ ),*

$$\mathbb{E}[X] = (\#S_\ell), \quad \text{Var}[X] \leq 2m^\ell \cdot \mathbb{E}[X].$$

## The Estimator Algorithm for Arbitrary Subgraphs

We now present our main estimator for the number of occurrences of an arbitrary subgraph  $H$  in  $G$ , denoted by  $(\#H)$ . Recall the decomposition  $\mathcal{D}(H) := \{\mathcal{C}_1, \dots, \mathcal{C}_o, \mathcal{S}_1, \dots, \mathcal{S}_s\}$  of  $H$  introduced in Section 3. Our algorithm creates a *subgraph-sampler tree*  $\mathcal{T}$  (a generalization of cycle-sampler and star-sampler trees in Definitions 6 and 9) and use it to estimate  $(\#H)$ . We define the subgraph-sampler tree  $\mathcal{T}$  and the algorithm `subgraph-sampler`( $G, H$ ) that creates it simultaneously:

**Subgraph-Sampler Tree.** The subgraph-sampler tree  $\mathcal{T}$  is a  $z$ -level tree for  $z := (2o + 2s)$  returned by `subgraph-sampler`( $G, H$ ). The algorithm constructs  $\mathcal{T}$  as follows.

*Sampling Odd Cycles.* In `subgraph-sampler`( $G, H$ ), we run `odd-cycle-sampler`( $G, \mathcal{C}_1$ ) and initiate  $\mathcal{T}$  to be its output cycle-sampler tree. For every (current) leaf-node  $\alpha$  of  $\mathcal{T}$ , we run `odd-cycle-sampler`( $G, \mathcal{C}_2$ ) *independently* to obtain a cycle-sampler tree  $\mathcal{T}_\alpha$  (we say that  $\alpha$  *started* the sampling of  $\mathcal{T}_\alpha$ ). We then extend the tree  $\mathcal{T}$  with two new layers by connecting each leaf-node  $\alpha$  to the root of  $\mathcal{T}_\alpha$  that started its sampling. This creates a

4-level tree  $\mathcal{T}$ . We continue like this for  $o$  steps, each time appending the tree obtained by `odd-cycle-sampler`( $G, \mathcal{C}_j$ ) for  $j \in [o]$ , to the (previous) leaf-node that started this sampling. This results in a  $(2o)$ -level tree. Note that the nodes in the tree  $\mathcal{T}$  can have different degrees as the number of leaf-nodes in the cycle-sampler tree is not necessarily the same always (not even for two different trees associated with one single  $\mathcal{C}_j$  through different calls to `odd-cycle-sampler`( $G, \mathcal{C}_j$ )).

*Sampling Stars.* Once we iterated over all odd cycles of  $\mathcal{D}(H)$ , we switch to processing stars  $\mathcal{S}_1, \dots, \mathcal{S}_s$ . The approach is identical to the previous part. Let  $\alpha$  be a (current) leaf-node of  $\mathcal{T}$ . We run `star-sampler`( $G, \mathcal{S}_1$ ) to obtain a star-sampler tree  $\mathcal{T}_\alpha$  and connect  $\alpha$  to  $\mathcal{T}_\alpha$  to extend the levels of tree by 2 more. We continue like this for  $s$  steps, each time appending the tree obtained by `star-sampler`( $G, \mathcal{S}_j$ ) for  $j \in [s]$ , to the (former) leaf-node that started this sampling. This results in a  $z$ -level tree  $\mathcal{T}$ . Note that all nodes added when sampling stars have exactly one child-node (except for the leaf-nodes) as by Definition 9, star-sampler trees always contain only two nodes.

*Labels and Values.* Each node  $\alpha$  of  $\mathcal{T}$  is again given two attributes, `label`[ $\alpha$ ] and `value`[ $\alpha$ ], which are defined to be exactly the same attributes in the corresponding cycle-sampler or star-sampler tree that was used to define these nodes (recall that each node of  $\mathcal{T}$  is “copied” from a node in either a cycle-sampler or a star-sampler tree). Finally, for each root-to-leaf path  $\mathcal{P}$  in  $\mathcal{T}$ , we define `label`[ $\mathcal{P}$ ] :=  $\bigcup_{\alpha \in \mathcal{P}} \text{label}[\alpha]$  and `value`[ $\mathcal{P}$ ] :=  $\prod_{\alpha \in \mathcal{P}} \text{value}[\alpha]$ . In particular, `label`[ $\mathcal{P}$ ] :=  $((e_1, w_1), \dots, (e_o, w_o), (v_1, \mathbf{w}_1), \dots, (v_s, \mathbf{w}_s))$  by definition of labels of cycle-sampler and star-sampler trees. As such `label`[ $\mathcal{P}$ ] is a representation of the subgraph  $H$  as defined in Section 3.2. By making  $O(1)$  additional pair-queries to query all the remaining edges of this representation of  $H$  we determine if `label`[ $\mathcal{P}$ ] forms a copy of  $H$ .

This concludes the description of `subgraph-sampler`( $G, H$ ) and its output subgraph-sampler tree  $\mathcal{T}$ . We bound the query complexity of the algorithm in the following lemma (the proof is postponed to the full version [4]).

► **Lemma 11.** *The expected query complexity/ running time of `subgraph-sampler` is  $O(1)$ .*

We are now ready to present our estimator algorithm using `subgraph-sampler` and the subgraph-sampler tree  $\mathcal{T}$  it outputs.

**An Estimator for Arbitrary Subgraphs.** Note that as before the subgraph-sampler tree  $\mathcal{T}$  itself is a random variable depending on the randomness of `subgraph-sampler`. For any root-to-leaf path  $\mathcal{P}_i := \alpha_1, \dots, \alpha_z$  of  $\mathcal{T}$ , we define the random variable  $X_i$  such that  $X_i := \text{value}[\mathcal{P}_i]$  iff `label`[ $\mathcal{P}_i$ ] forms a copy of  $H$  in  $G$  and otherwise  $X_i := 0$ . We further define  $Y := (\frac{1}{t} \sum_{i=1}^t X_i)$ , where  $t$  is the number of leaf-nodes of  $\mathcal{T}$  (which itself is a random variable). These random variables can all be computed from  $\mathcal{T}$  and `subgraph-sampler` with at most  $O(1)$  further pair-queries per each root-to-leaf path  $\mathcal{P}$  of the tree to determine if indeed `label`[ $\mathcal{P}$ ] forms a copy of  $H$  in  $G$  or not. As such, query complexity and runtime of this algorithm is proportional to `subgraph-sampler` (which in expectation is  $O(1)$  by Lemma 11). In the following two lemmas, we show that  $Y$  is a low-variance unbiased estimator of  $(\#H)$ .

**Notation.** For any node  $\alpha$  in  $\mathcal{T}$ , we use  $\mathcal{T}_\alpha$  to denote the sub-tree of  $\mathcal{T}$  rooted at  $\alpha$ . For a leaf-node  $\alpha$ , we define a random variable  $Y_\alpha$  which is `value`[ $\alpha$ ] iff for the root-to-leaf path  $\mathcal{P}$  ending in  $\alpha$ , `label`[ $\mathcal{P}$ ] forms a copy of  $H$  in  $G$  and otherwise  $Y_\alpha$  is 0. For an internal node  $\alpha$  in  $\mathcal{T}$  with  $t$  child-nodes  $\alpha_1, \dots, \alpha_t$ , we define  $Y_\alpha = \text{value}[\alpha] \cdot (\frac{1}{t} \cdot \sum_{i=1}^t Y_i)$ . It is easy to verify that  $Y_{\alpha_r}$  for the root  $\alpha_r$  of  $\mathcal{T}$  is the same as the estimator random variable  $Y$  defined earlier. Furthermore, for a node  $\alpha$  in level  $\ell$  of  $\mathcal{T}$ , we define  $\mathbf{L}_\alpha := (\text{label}[\alpha_1], \text{label}[\alpha_2], \dots, \text{label}[\alpha_{\ell-1}])$ , where  $\alpha_1, \dots, \alpha_{\ell-1}$  forms the path from the root of  $\mathcal{T}$  to the parent of  $\alpha$ .

We analyze the expected value and the variance of the estimator.

► **Lemma 12.** *For  $Y$  in  $\text{subgraph-sampler}(G, H)$ ,  $\mathbb{E}[Y] = (\#H)$ .*

**Proof.** We prove this inductively by showing that for any node  $\alpha$  in an *odd layer* of  $\mathcal{T}$ ,  $\mathbb{E}[Y_\alpha \mid \mathbf{L}_\alpha] = (\#H \mid \mathbf{L}_\alpha)$ , where  $(\#H \mid \mathbf{L}_\alpha)$  denotes the number of copies of  $H$  in  $G$  that contain the vertices and edges specified by  $\mathbf{L}_\alpha$  (according to the decomposition  $\mathcal{D}(H)$ ).  $\mathbb{E}[Y_\alpha \mid \mathbf{L}_\alpha]$  measures the value of  $Y_\alpha$  after we fix the rest of the tree  $\mathcal{T}$  and let the sub-tree  $\mathcal{T}_\alpha$  be chosen randomly as in  $\text{subgraph-sampler}$ .

The base case of the induction, i.e., for vertices in the last odd layer of  $\mathcal{T}$  follows exactly as in the proofs of Lemmas 8 and 10 (as will also become evident shortly) and hence we do not repeat it here. We now prove the induction hypothesis. Fix a vertex  $\alpha$  in an odd layer  $\ell$ . We consider two cases based on whether  $\ell < 2o$  (hence  $\alpha$  is root of a cycle-sampler tree) or  $\ell > 2o$  (hence  $\alpha$  is root of a star-sampler tree).

**Case of  $\ell < 2o$ .** In this case, the sub-tree  $\mathcal{T}_\alpha$  in the next two levels is a cycle-sampler tree,

$$\begin{aligned} \mathbb{E}[Y_\alpha \mid \mathbf{L}_\alpha] &= \sum_e \Pr(\text{label}[\alpha] = e) \cdot \text{value}[\alpha] \cdot \left( \frac{1}{t_e} \sum_{i=1}^{t_e} \mathbb{E}[Y_{\alpha_i} \mid \mathbf{L}_\alpha, e] \right) \\ &\hspace{15em} \text{(here, } \alpha_i \text{'s are child-nodes of } \alpha \text{)} \\ &= \sum_e \frac{1}{t_e} \sum_{i=1}^{t_e} \mathbb{E}[Y_{\alpha_i} \mid \mathbf{L}_\alpha, e] \\ &\hspace{15em} \text{(as by definition, } \text{value}[\alpha] = \Pr(\text{label}[\alpha] = e)^{-1} \text{)} \end{aligned}$$

Note that each  $\alpha_i$  has exactly one child-node, denoted by  $\beta_i$ . As such,

$$\begin{aligned} \mathbb{E}[Y_\alpha \mid \mathbf{L}_\alpha] &= \sum_e \frac{1}{t_e} \sum_{i=1}^{t_e} \mathbb{E}[Y_{\alpha_i} \mid \mathbf{L}_\alpha, e] \\ &= \sum_e \frac{1}{t_e} \sum_{i=1}^{t_e} \sum_w \Pr(\text{label}[\alpha_i] = w) \cdot \text{value}[\alpha_i] \cdot \mathbb{E}[Y_{\beta_i} \mid \mathbf{L}_\alpha, e, w] \\ &= \sum_e \frac{1}{t_e} \sum_{i=1}^{t_e} \sum_w \mathbb{E}[Y_{\beta_i} \mid \mathbf{L}_{\beta_i}] \\ &\hspace{10em} \text{(by definition } \text{value}[\alpha_i] = \Pr(\text{label}[\alpha_i] = w)^{-1} \text{ and } \mathbf{L}_{\beta_i} = \mathbf{L}_\alpha, (e, w) \text{)} \\ &= \sum_e \frac{1}{t_e} \sum_{i=1}^{t_e} \sum_w (\#H \mid \mathbf{L}_{\beta_i}) = \sum_e \frac{1}{t_e} \sum_{i=1}^{t_e} \sum_w (\#H \mid \mathbf{L}_\alpha, (e, w)) \\ &\hspace{10em} \text{(by induction hypothesis for odd-layer nodes } \beta_i \text{'s)} \\ &= \sum_e \sum_w (\#H \mid \mathbf{L}_\alpha, (e, w)) = (\#H \mid \mathbf{L}_\alpha). \end{aligned}$$

This concludes the proof of induction hypothesis in this case.

**Case of  $\ell > 2o$ .** In this case, the sub-tree  $\mathcal{T}_\alpha$  in the next two levels is a star-sampler tree.

By the same analogy made in the proof of the previous part and Lemma 8, the proof of this part also follows directly from the proof of Lemma 10 for star-sampler trees.

We can now finalize the proof of Lemma 12, by noting that for the root  $\alpha_r$  of  $\mathcal{T}$ ,  $\mathbf{L}_{\alpha_r}$  is the empty-set and hence,  $\mathbb{E}[Y] = \mathbb{E}[Y_{\alpha_r} \mid \mathbf{L}_{\alpha_r}]$ , which by induction is equal to  $(\#H)$ . ◀

► **Lemma 13.** For  $Y$  in  $\text{subgraph-sampler}(G, H)$ ,  $\text{Var}[Y] = O(m^{\rho(H)}) \cdot \mathbb{E}[Y]$ .

**Proof.** We bound  $\text{Var}[Y]$  using a similar inductive proof as in Lemma 12. Recall the parameters  $\rho_1^C, \dots, \rho_o^C$  and  $\rho_1^S, \dots, \rho_s^S$  associated respectively with the cycles  $\mathcal{C}_1, \dots, \mathcal{C}_o$  and stars  $\mathcal{S}_1, \dots, \mathcal{S}_s$  of the decomposition  $\mathcal{D}(H)$ . For simplicity of notation, for any  $i \in [o + s]$ , we define  $\rho_{i+}$  as follows:

$$\text{for all } i \leq o, \rho_{i+} := \sum_{j=i}^o \rho_j^C + \sum_{j=1}^s \rho_j^S, \quad \text{for all } o < i \leq o + s, \rho_{i+} := \sum_{j=i-o}^s \rho_j^S.$$

We inductively show that, for any node  $\alpha$  in an *odd layer*  $2\ell - 1$  of  $\mathcal{T}$ ,

$$\text{Var}[Y_\alpha \mid \mathbf{L}_\alpha] \leq 2^{2z-2\ell} \cdot m^{\rho_{\ell+}} \cdot (\#H \mid \mathbf{L}_\alpha),$$

where  $(\#H \mid \mathbf{L}_\alpha)$  denotes the number of copies of  $H$  in  $G$  that contain the vertices and edges specified by  $\mathbf{L}_\alpha$  (according to the decomposition  $\mathcal{D}(H)$ ).

The induction is from the leaf-nodes of the tree to the root. The base case of the induction, i.e., for vertices in the last odd layer of  $\mathcal{T}$  follows exactly as in the proofs of Lemmas 8 and 10 (as will also become evident shortly) and hence we do not repeat it here. We now prove the induction hypothesis. Fix a vertex  $\alpha$  in an odd layer  $2\ell - 1$ . We consider two cases based on whether  $\ell \leq o$  (hence  $\alpha$  is root of a cycle-sampler tree) or  $\ell > o$  (hence  $\alpha$  is root of a star-sampler tree).

**Case of  $\ell \leq o$ .** In this case, the sub-tree  $\mathcal{T}_\alpha$  in the next two levels is a cycle-sampler tree corresponding to the odd cycle  $\mathcal{C}_\ell$  of  $\mathcal{D}(H)$ . Let the number of edges in  $\mathcal{C}_\ell$  be  $(2k + 1)$  (i.e.,  $\mathcal{C}_\ell = \mathcal{C}_{2k+1}$ ) Let  $e$  denote the label of the  $\alpha$ . By the law of total variance in Eq. (1)

$$\text{Var}[Y_\alpha \mid \mathbf{L}_\alpha] = \mathbb{E}[\text{Var}[Y_\alpha \mid e] \mid \mathbf{L}_\alpha] + \text{Var}[\mathbb{E}[Y_\alpha \mid e] \mid \mathbf{L}_\alpha]. \quad (7)$$

We start by bounding the second term in Eq. (7) which is easier. By the inductive proof of Lemma 12, we also have,  $\mathbb{E}[Y_\alpha \mid \mathbf{L}_\alpha, e] = (\#H \mid \mathbf{L}_\alpha, e)$ . As such,

$$\begin{aligned} \text{Var}[\mathbb{E}[Y_\alpha \mid e] \mid \mathbf{L}_\alpha] &= \text{Var}[(\#H \mid \mathbf{L}_\alpha, e) \mid \mathbf{L}_\alpha] \leq \mathbb{E}[(\#H \mid \mathbf{L}_\alpha, e)^2 \mid \mathbf{L}_\alpha] \\ &= \sum_e \Pr(\text{label}[\alpha] = e) \cdot (\#H \mid \mathbf{L}_\alpha, e)^2 = \frac{1}{m^k} \sum_e (\#H \mid \mathbf{L}_\alpha, e)^2 \\ &\quad (\Pr(\text{label}[\alpha] = e) = 1/m^k \text{ by definition of odd-cycle-sampler}) \\ &\leq \frac{1}{m^k} \left( \sum_e (\#H \mid \mathbf{L}_\alpha, e) \right)^2 = \frac{1}{m^k} (\#H \mid \mathbf{L}_\alpha)^2 \\ &\leq m^{\rho_{\ell+}} \cdot (\#H \mid \mathbf{L}_\alpha). \end{aligned} \quad (8)$$

The reason behind the last equality is that  $(\#H \mid \mathbf{L}_\alpha)$  is at most equal to the number of copies of the subgraph of  $H$  consisting of  $\mathcal{C}_\ell, \dots, \mathcal{C}_o, \mathcal{S}_1, \dots, \mathcal{S}_s$ , which by Lemma 5 is at most  $m^{\rho_{\ell+}}$  by definition of  $\rho_{\ell+}$ . We now bound the first and the main term in Eq. (7),

$$\begin{aligned} \mathbb{E}[\text{Var}[Y_\alpha \mid e] \mid \mathbf{L}_\alpha] &= \sum_e \Pr(\text{label}[\alpha] = e) \cdot \text{Var}[Y_\alpha \mid e, \mathbf{L}_\alpha] \\ &= \sum_e \frac{1}{m^k} \cdot m^{2k} \cdot \frac{1}{t_e^2} \cdot \sum_{i=1}^{t_e} \text{Var}[Y_{\alpha_i} \mid e, \mathbf{L}_\alpha], \end{aligned}$$

(here  $\alpha_i$ 's are child-nodes of  $\alpha$ )

where the final equality holds because  $Y_{\alpha_i}$ 's are independent conditioned on  $\mathbf{e}, \mathbf{L}_\alpha$  and since  $Y_\alpha$  is by definition  $m^k$  times the average of  $Y_{\alpha_i}$ 's. Moreover, note that distribution of all  $Y_{\alpha_i}$ 's are the same. Hence, by canceling the terms,

$$\mathbb{E}[\text{Var}[Y_\alpha | \mathbf{e}] | \mathbf{L}_\alpha] = m^k \cdot \sum_{\mathbf{e}} \frac{1}{t_{\mathbf{e}}} \cdot \text{Var}[Y_{\alpha_1} | \mathbf{e}, \mathbf{L}_\alpha], \quad (9)$$

We thus only need to bound  $\text{Var}[Y_{\alpha_1} | \mathbf{e}, \mathbf{L}_\alpha]$ . Recall that  $\alpha_1$  corresponds to a leaf-node in a cycle-sampler tree and hence its label is a vertex  $w$  from the neighborhood of  $u_{\mathbf{e}}^*$  as defined in `odd-cycle-sampler`. We again use the law of total variance in Eq. (1) to obtain,

$$\text{Var}[Y_{\alpha_1} | \mathbf{e}, \mathbf{L}_\alpha] = \mathbb{E}[\text{Var}[Y_{\alpha_1} | w] | \mathbf{e}, \mathbf{L}_\alpha] + \text{Var}[\mathbb{E}[Y_{\alpha_1} | w] | \mathbf{e}, \mathbf{L}_\alpha] \quad (10)$$

For the first term,

$$\begin{aligned} \mathbb{E}[\text{Var}[Y_{\alpha_1} | w] | \mathbf{e}, \mathbf{L}_\alpha] &= \sum_{w \in N(u_{\mathbf{e}}^*)} \Pr(\text{label}[\alpha_1] = w) \cdot \text{Var}[Y_{\alpha_1} | w, \mathbf{e}, \mathbf{L}_\alpha] \\ &= \sum_w \frac{1}{d_{\mathbf{e}}^*} \cdot (d_{\mathbf{e}}^*)^2 \cdot \text{Var}[Y_{\beta_1} | w, \mathbf{e}, \mathbf{L}_\alpha], \end{aligned}$$

where  $\beta_1$  is the unique child-node of  $\alpha_1$  and so  $Y_{\alpha_1} = \text{value}[\alpha_1] \cdot Y_{\beta_1}$ , while conditioned on  $\mathbf{e}$ ,  $\text{value}[\alpha_1] = d_{\mathbf{e}}^*$ . Moreover, as  $\mathbf{L}_{\beta_1} = (\mathbf{L}_\alpha, \mathbf{e}, w)$ , and by canceling the terms,

$$\begin{aligned} \mathbb{E}[\text{Var}[Y_{\alpha_1} | w] | \mathbf{e}, \mathbf{L}_\alpha] &= \sum_w d_{\mathbf{e}}^* \cdot \text{Var}[Y_{\beta_1} | \mathbf{L}_{\beta_1}] \\ &\leq \sum_w d_{\mathbf{e}}^* \cdot 2^{2z-2\ell-2} \cdot m^{\rho(\ell+1)+} \cdot (\#H | \mathbf{L}_{\beta_1}), \end{aligned} \quad (11)$$

where the inequality is by induction hypothesis for the odd-level node  $\beta_1$ . We now bound the second term in Eq. (10) as follows,

$$\begin{aligned} \text{Var}[\mathbb{E}[Y_{\alpha_1} | w] | \mathbf{e}, \mathbf{L}_\alpha] &\leq \mathbb{E}\left[\left(\mathbb{E}[Y_{\alpha_1} | w]\right)^2 | \mathbf{e}, \mathbf{L}_\alpha\right] \\ &= \sum_w \Pr(\text{label}[\alpha_1] = w) \cdot \left(\mathbb{E}[Y_{\alpha_1} | w, \mathbf{e}, \mathbf{L}_\alpha]\right)^2 \\ &= \sum_w \frac{1}{d_{\mathbf{e}}^*} \cdot (d_{\mathbf{e}}^*)^2 \cdot \left(\mathbb{E}[Y_{\beta_1} | w, \mathbf{e}, \mathbf{L}_\alpha]\right)^2 \\ &= \sum_w d_{\mathbf{e}}^* \cdot \left(\mathbb{E}[Y_{\beta_1} | \mathbf{L}_{\beta_1}]\right)^2 = \sum_w d_{\mathbf{e}}^* \cdot (\#H | \mathbf{L}_{\beta_1})^2 \\ &\leq \sum_w d_{\mathbf{e}}^* \cdot m^{\rho(\ell+1)+} \cdot (\#H | \mathbf{L}_{\beta_1}). \end{aligned} \quad (12)$$

Here, the second to last equality holds by the inductive proof of Lemma 12, and the last equality is because  $(\#H | \mathbf{L}_{\beta_1}) \leq m^{\rho(\ell+1)+}$  by Lemma 5, as  $(\#H | \mathbf{L}_{\beta_1})$  is at most equal to the total number of copies of a subgraph of  $H$  on  $\mathcal{C}_{\ell+1}, \dots, \mathcal{C}_o, \mathcal{S}_1, \dots, \mathcal{S}_s$  (and by definition of  $\rho_{(\ell+1)+}$ ). We now plug in Eq. (11) and Eq. (12) in Eq. (10),

$$\text{Var}[Y_{\alpha_1} | \mathbf{e}, \mathbf{L}_\alpha] \leq \sum_w d_{\mathbf{e}}^* \cdot (2^{2z-2\ell-2} \cdot m^{\rho(\ell+1)+} \cdot (\#H | \mathbf{L}_{\beta_1}) + m^{\rho(\ell+1)+} \cdot (\#H | \mathbf{L}_{\beta_1})).$$

We now in turn plug this in Eq. (9),

$$\begin{aligned}
& \mathbb{E} [\text{Var} [Y_\alpha \mid e] \mid \mathbf{L}_\alpha] \\
& \leq m^k \sum_e \frac{1}{t_e} \sum_w d_e^* \cdot (2^{2z-2\ell-2} \cdot m^{\rho(\ell+1)+} \cdot (\#H \mid \mathbf{L}_{\beta_1}) + m^{\rho(\ell+1)+} \cdot (\#H \mid \mathbf{L}_{\beta_1})) \\
& \leq m^k \sqrt{m} \cdot \sum_e \sum_w 2^{2z-2\ell-1} \cdot m^{\rho(\ell+1)+} \cdot (\#H \mid \mathbf{L}_{\beta_1}) \quad (\text{as } t_e \geq d_e^*/\sqrt{m}) \\
& \leq 2^{2z-2\ell-1} \cdot m^{\rho_{\ell+}} \cdot \sum_e \sum_w (\#H \mid \mathbf{L}_{\beta_1}) \\
& \quad (\text{as } \rho_\ell^C = k + 1/2 \text{ and } \rho_{\ell+} = \rho_\ell^C + \rho_{(\ell+1)+} \text{ by definition}) \\
& = 2^{2z-2\ell-1} \cdot m^{\rho_{\ell+}} \cdot (\#H \mid \mathbf{L}_\alpha). \quad (\text{as } \mathbf{L}_{\beta_1} = (\mathbf{L}_\alpha, e, w))
\end{aligned}$$

Finally, by plugging in this and Eq. (8) in Eq. (7),

$$\begin{aligned}
\text{Var} [Y_\alpha \mid \mathbf{L}_\alpha] &= 2^{2z-2\ell-1} \cdot m^{\rho_{\ell+}} \cdot (\#H \mid \mathbf{L}_\alpha) + m^{\rho_{\ell+}} \cdot (\#H \mid \mathbf{L}_\alpha) \\
&\leq 2^{2z-2\ell} \cdot m^{\rho_{\ell+}} \cdot (\#H \mid \mathbf{L}_\alpha),
\end{aligned}$$

finalizing the proof of induction step in this case. We again remark that this proof closely followed the proof for the variance of the estimator for cycle-sampler tree in Lemma 8.

**Case of  $\ell > o$ .** In this case, the sub-tree  $\mathcal{T}_\alpha$  in the next two levels is a star-sampler tree. By the same analogy made in the proof of the previous case and Lemma 8, the proof of this part also follows the proof of Lemma 10 for star-sampler trees. We hence omit the details.

To conclude, we have that  $\text{Var} [Y] = \text{Var} [Y_{\alpha_r} \mid \mathbf{L}_{\alpha_r}] = O(m^{\rho(H)} \cdot (\#H)) = O(m^{\rho(H)}) \cdot \mathbb{E} [Y]$  as  $Y = Y_{\alpha_r}$  for the root  $\alpha_r$  of  $\mathcal{T}$ ,  $\mathbf{L}_{\alpha_r} = \emptyset$ ,  $(\#H) = \mathbb{E} [Y]$  by Lemma 12, and  $z = O(1)$ .  $\blacktriangleleft$

## 4.2 An Algorithm for Estimating Occurrences of Arbitrary Subgraphs

We now use our estimator algorithm from the previous section to design our algorithm for estimating the occurrences of an arbitrary subgraph  $H$  in  $G$ . In the following theorem, we assume that the algorithm has knowledge of  $m$  and also a lower bound on the value of  $\#H$ ; these assumptions can be lifted easily as we describe afterwards.

**► Theorem 14.** *There exists a sublinear time algorithm that uses degree, neighbor, pair, and edge sample queries and given a precision parameter  $\varepsilon \in (0, 1)$ , an explicit access to a constant-size graph  $H(V_H, E_H)$ , a query access to the input graph  $G(V, E)$ , the number of edges  $m$  in  $G$ , and a lower bound  $h \leq \#H$ , with high probability outputs a  $(1 \pm \varepsilon)$ -approximation to  $\#H$  using  $O\left(\min\left\{m, \frac{m^{\rho(H)}}{h} \cdot \frac{\log n}{\varepsilon^2}\right\}\right)$  queries and  $O\left(\frac{m^{\rho(H)}}{h} \cdot \frac{\log n}{\varepsilon^2}\right)$  time, in the worst-case.*

**Proof.** Fix a sufficiently large constant  $c > 0$ . We run `subgraph-sampler`( $G, H$ ) for  $k := \frac{c \cdot m^{\rho(H)}}{\varepsilon^2 \cdot h}$  time independently in parallel to obtain estimates  $Y_1, \dots, Y_k$  and let  $Z := \frac{1}{k} \sum_{i=1}^k Y_i$ . By Lemma 12,  $\mathbb{E} [Z] = (\#H)$ . Since  $Y_i$ 's are independent, we also have

$$\text{Var} [Z] = \frac{1}{k^2} \sum_{i=1}^k \text{Var} [Y_i] \leq \frac{1}{k} \cdot O(m^{\rho(H)}) \cdot \mathbb{E} [Z] \leq \frac{\varepsilon^2}{10} \cdot \mathbb{E} [Z]^2,$$

by Lemma 13, and by choosing the constant  $c$  sufficiently larger than the constant in the  $O$ -notation of this lemma, together with the fact that  $h \leq (\#H) = \mathbb{E} [Z]$ . By Chebyshev's inequality (Proposition 2),  $\Pr(|Z - \mathbb{E} [Z]| \geq \varepsilon \cdot \mathbb{E} [Z]) \leq \frac{\text{Var} [Z]}{\varepsilon^2 \cdot \mathbb{E} [Z]^2} \leq \frac{1}{10}$ , by the bound above on the variance. This means that with probability 0.9, this algorithm outputs a  $(1 \pm \varepsilon)$ -approximation of  $\#H$ . Moreover, the expected query complexity and running time of this

algorithm is  $O(k)$  by Lemma 11, which is  $O(\frac{m^{\rho(H)}}{\varepsilon^2})$  (if  $k \geq m$ , we simply query all edges of the graph and solve the problem using an offline enumeration algorithm). To extend this result to a high probability bound and also making the guarantee of query complexity and run-time in the worst-case, we simply run this algorithm  $O(\log n)$  times in parallel and stop each execution that uses more than 10 times queries than the expectation. ◀

The algorithm in Theorem 14 assumes the knowledge of  $h$  which is a lower bound on  $(\#H)$ . However, this assumption can be easily removed by making a geometric search on  $h$  starting from  $m^{\rho(H)}/2$  which is (approximately) the largest value for  $(\#H)$  all the way down to 1 in factors of 2, and stopping the search once the estimates returned for a guess of  $h$  became consistent with  $h$  itself. This only increases the query complexity and runtime of the algorithm by polylog( $n$ ) factors. As this part is quite standard, we omit the details and instead refer the interested reader to [16, 18]. This concludes the proof of our main result in Theorem 1 from the introduction.

### 4.3 Extension to the Database Join Size Estimation Problem

The database join size estimation for binary relations can be modeled by the subgraph estimation problem where the subgraph  $H$  and the underlying graph  $G$  are additionally *edge-colored* and we are only interested in counting the copies of  $H$  in  $G$  with matching colors on the edges. In this abstraction, the edges of the graph  $G$  correspond to the entries of the database, and the color of edges determine the relation of the entry.

We formalize this variant of the subgraph counting problem in the following. In the *colorful* subgraph estimation problem, we are given a subgraph  $H(V_H, E_H)$  with a coloring function  $c_H : E_H \rightarrow \mathbb{N}$  and query access to a graph  $G(V, E)$  along with a coloring function  $c_G : E \rightarrow \mathbb{N}$ . The set of allowed queries to  $G$  contains the degree queries, pair queries, neighbor queries, and edge-sample queries as before, with a simple change that whenever we query an edge (through the last three types of queries), the color of the edge according to  $c_G$  is also revealed to the algorithm. Our goal is to estimate the number of copies of  $H$  in  $G$  with matching colors, i.e., the *colorful* copies of  $H$ .

It is immediate to verify that our algorithm in this section can be directly applied to the colorful subgraph estimation problem with the only difference that when testing whether a subgraph forms a copy of  $H$  in  $G$ , we in fact check whether this subgraph forms a colorful copy of  $H$  in  $G$  instead. The analysis of this new algorithm is exactly as in the case of the original algorithm with the only difference that we switch the parameter  $\#H$  to  $\#H_c$  that only counts the number of copies of  $H$  with the same colors in  $G$ . To summarize, we obtain an algorithm with  $O^*(\frac{m^{\rho(H)}}{\#H_c})$  query and time complexity for the colorful subgraph counting problem, which can in turn solves the database join size estimation problem for binary relations.

## 5 Lower Bounds

We present two lower bounds that demonstrate the optimality of Theorem 1 in different scenarios. Our first lower bound establishes tight bounds for counting *odd cycles*.

► **Theorem 15.** *For any  $k \geq 1$ , any algorithm  $\mathcal{A}$  that can output any multiplicative-approximation to the number of copies of the odd cycle  $C_{2k+1}$  in a given graph  $G(V, E)$  with probability at least  $2/3$  requires  $\Omega(\frac{m^{k+\frac{1}{2}}}{\#C_{2k+1}})$  queries to  $G$ .*

Theorem 15 implies that in addition to cliques (that were previously proved [19]; see also [16, 18]), our algorithm in Theorem 1 also achieve optimal bounds for odd cycles.

Our next lower bound targets the more general problem of database join size estimation for which we argued that our Theorem 1 continues to hold. We show that for this more general problem, our algorithm in Theorem 1 is in fact optimal for all choices of subgraph  $H$ .

► **Theorem 16.** *For any subgraph  $H(V_H, E_H)$  which contains at least one edge, suppose  $\mathcal{A}$  is an algorithm for the colorful subgraph estimation problem that given  $H$ , a coloring  $c_H : E_H \rightarrow \mathbb{N}$ , and query access to  $G(V, E)$  with  $m$  edges and coloring function  $c_G : E \rightarrow \mathbb{N}$ , can output a multiplicative-approximation to the number of colorful copies of  $H$  in  $G$  with probability at least  $2/3$ . Then,  $\mathcal{A}$  requires  $\Omega(\frac{m^{p(H)}}{\#H_c})$  queries, where  $\#H_c$  is the number of colorful copies of  $H$  in  $G$ . The lower bound continues to hold even if the number of colors used by  $c_H$  and  $c_G$  is at most two.*

The proofs of Theorems 15 and 16 are postponed to the full version of the paper [4].

---

## References

- 1 Nesreen K. Ahmed, Jennifer Neville, and Ramana Rao Kompella. Network Sampling: From Static to Streaming Graphs. *TKDD*, 8(2):7:1–7:56, 2013.
- 2 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.
- 3 Noga Alon. On the number of subgraphs of prescribed type of graphs with a given number of edges. *Israel Journal of Mathematics*, 1981.
- 4 Sepehr Assadi, Michael Kapralov, and Sanjeev Khanna. A Simple Sublinear-Time Algorithm for Counting Arbitrary Subgraphs via Edge Sampling. *arXiv*, abs/1811.07780, 2018. [arXiv:1811.07780](https://arxiv.org/abs/1811.07780).
- 5 Albert Atserias, Martin Grohe, and Dániel Marx. Size Bounds and Query Plans for Relational Joins. In *49th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2008, October 25–28, 2008, Philadelphia, PA, USA*, pages 739–748. IEEE Computer Society, 2008. doi:10.1109/FOCS.2008.43.
- 6 Ziv Bar-Yossef, Ravi Kumar, and D. Sivakumar. Reductions in streaming algorithms, with an application to counting triangles in graphs. In *Proceedings of the Thirteenth Annual ACM-SIAM Symposium on Discrete Algorithms, January 6–8, 2002, San Francisco, CA, USA.*, pages 623–632, 2002.
- 7 Suman K. Bera and Amit Chakrabarti. Towards Tighter Space Bounds for Counting Triangles and Other Substructures in Graph Streams. In *34th Symposium on Theoretical Aspects of Computer Science, STACS 2017, March 8–11, 2017, Hannover, Germany*, pages 11:1–11:14, 2017.
- 8 E. Bloedorn, N. Rothleder, D. DeBarr, and L. Rosen. Relational Graph Analysis with Real-World Constraints: An Application in IRS Tax Fraud Detection. In *AAAI*, 2005.
- 9 Vladimir Braverman, Rafail Ostrovsky, and Dan Vilenchik. How Hard Is Counting Triangles in the Streaming Model? In *Automata, Languages, and Programming - 40th International Colloquium, ICALP 2013, Riga, Latvia, July 8–12, 2013, Proceedings, Part I*, pages 244–254, 2013.
- 10 Luciana S. Buriol, Gereon Frahling, Stefano Leonardi, Alberto Marchetti-Spaccamela, and Christian Sohler. Counting triangles in data streams. In *Proceedings of the Twenty-Fifth ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems, June 26–28, 2006, Chicago, Illinois, USA*, pages 253–262, 2006.



- 11 S. Burt. Structural Holes and Good Ideas. *The American Journal of Sociology*, 110(2):349–399, 2004. doi:10.2307/3568221.
- 12 Bernard Chazelle, Ronitt Rubinfeld, and Luca Trevisan. Approximating the Minimum Spanning Tree Weight in Sublinear Time. *SIAM J. Comput.*, 34(6):1370–1379, 2005.
- 13 Graham Cormode and Hossein Jowhari. A second look at counting triangles in graph streams (corrected). *Theor. Comput. Sci.*, 683:22–30, 2017.
- 14 Artur Czumaj, Funda Ergün, Lance Fortnow, Avner Magen, Ilan Newman, Ronitt Rubinfeld, and Christian Sohler. Approximating the Weight of the Euclidean Minimum Spanning Tree in Sublinear Time. *SIAM J. Comput.*, 35(1):91–109, 2005.
- 15 Artur Czumaj and Christian Sohler. Estimating the weight of metric minimum spanning trees in sublinear-time. In *Proceedings of the 36th Annual ACM Symposium on Theory of Computing, Chicago, IL, USA, June 13-16, 2004*, pages 175–183, 2004.
- 16 Talya Eden, Amit Levi, Dana Ron, and C. Seshadhri. Approximately Counting Triangles in Sublinear Time. In *IEEE 56th Annual Symposium on Foundations of Computer Science, FOCS 2015, Berkeley, CA, USA, 17-20 October, 2015*, pages 614–633, 2015.
- 17 Talya Eden, Dana Ron, and C. Seshadhri. Sublinear Time Estimation of Degree Distribution Moments: The Degeneracy Connection. In *44th International Colloquium on Automata, Languages, and Programming, ICALP 2017, July 10-14, 2017, Warsaw, Poland*, pages 7:1–7:13, 2017.
- 18 Talya Eden, Dana Ron, and C. Seshadhri. On approximating the number of k-cliques in sublinear time. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2018, Los Angeles, CA, USA, June 25-29, 2018*, pages 722–734, 2018.
- 19 Talya Eden and Will Rosenbaum. Lower Bounds for Approximating Graph Parameters via Communication Complexity. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, APPROX/RANDOM 2018, August 20-22, 2018 - Princeton, NJ, USA*, pages 11:1–11:18, 2018.
- 20 Talya Eden and Will Rosenbaum. On Sampling Edges Almost Uniformly. In *1st Symposium on Simplicity in Algorithms, SOSA 2018, January 7-10, 2018, New Orleans, LA, USA*, pages 7:1–7:9, 2018.
- 21 Uriel Feige. On sums of independent random variables with unbounded variance, and estimating the average degree in a graph. In *Proceedings of the 36th Annual ACM Symposium on Theory of Computing, Chicago, IL, USA, June 13-16, 2004*, pages 594–603, 2004.
- 22 Ehud Friedgut and Jeff Kahn. On the number of copies of one hypergraph in another. *Israel Journal of Mathematics*, 1998.
- 23 Oded Goldreich. *Introduction to Property Testing*. Cambridge University Press, 2017.
- 24 Oded Goldreich and Dana Ron. Approximating average parameters of graphs. *Random Struct. Algorithms*, 32(4):473–493, 2008.
- 25 Mira Gonen, Dana Ron, and Yuval Shavitt. Counting Stars and Other Small Subgraphs in Sublinear Time. In *Proceedings of the Twenty-First Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2010, Austin, Texas, USA, January 17-19, 2010*, pages 99–116, 2010.
- 26 Avinatan Hassidim, Jonathan A. Kelner, Huy N. Nguyen, and Krzysztof Onak. Local Graph Partitions for Approximation and Testing. In *50th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2009, October 25-27, 2009, Atlanta, Georgia, USA*, pages 22–31, 2009.
- 27 Madhav Jha, C. Seshadhri, and Ali Pinar. A space efficient streaming algorithm for triangle counting using the birthday paradox. In *The 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD 2013, Chicago, IL, USA, August 11-14, 2013*, pages 589–597, 2013.

- 28 Hossein Jowhari and Mohammad Ghodsi. New Streaming Algorithms for Counting Triangles in Graphs. In *Computing and Combinatorics, 11th Annual International Conference, COCOON 2005, Kunming, China, August 16-29, 2005, Proceedings*, pages 710–716, 2005.
- 29 John Kallaugher, Michael Kapralov, and Eric Price. The Sketching Complexity of Graph and Hypergraph Counting. *CoRR*, abs/1808.04995. To appear in FOCS 2018., 2018.
- 30 John Kallaugher and Eric Price. A Hybrid Sampling Scheme for Triangle Counting. In *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2017, Barcelona, Spain, Hotel Porta Fira, January 16-19*, pages 1778–1797, 2017.
- 31 Daniel M. Kane, Kurt Mehlhorn, Thomas Sauerwald, and He Sun. Counting Arbitrary Subgraphs in Data Streams. In *Automata, Languages, and Programming - 39th International Colloquium, ICALP 2012, Warwick, UK, July 9-13, 2012, Proceedings, Part II*, pages 598–609, 2012.
- 32 Tali Kaufman, Michael Krivelevich, and Dana Ron. Tight Bounds for Testing Bipartiteness in General Graphs. *SIAM J. Comput.*, 33(6):1441–1483, 2004.
- 33 Ju-Sung Lee and Jürgen Pfeffer. Estimating Centrality Statistics for Complete and Sampled Networks: Some Approaches and Complications. In *48th Hawaii International Conference on System Sciences, HICSS 2015, Kauai, Hawaii, USA, January 5-8, 2015*, pages 1686–1695, 2015.
- 34 Jure Leskovec and Christos Faloutsos. Sampling from large graphs. In *Proceedings of the Twelfth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Philadelphia, PA, USA, August 20-23, 2006*, pages 631–636, 2006.
- 35 Andrew McGregor, Sofya Vorotnikova, and Hoa T. Vu. Better Algorithms for Counting Triangles in Data Streams. In *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, 2016.
- 36 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, October 2002.
- 37 Hung Q. Ngo, Ely Porat, Christopher Ré, and Atri Rudra. Worst-case Optimal Join Algorithms. *J. ACM*, 65(3):16:1–16:40, 2018.
- 38 Huy N. Nguyen and Krzysztof Onak. Constant-Time Approximation Algorithms via Local Improvements. In *49th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2008, October 25-28, 2008, Philadelphia, PA, USA*, pages 327–336, 2008.
- 39 Krzysztof Onak, Dana Ron, Michal Rosen, and Ronitt Rubinfeld. A near-optimal sublinear-time algorithm for approximating the minimum vertex cover size. In *Proceedings of the Twenty-Third Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2012, Kyoto, Japan, January 17-19, 2012*, pages 1123–1131, 2012.
- 40 Michal Parnas and Dana Ron. Approximating the minimum vertex cover in sublinear time and a connection to distributed algorithms. *Theor. Comput. Sci.*, 381(1-3):183–196, 2007.
- 41 Olivia Simpson, C. Seshadhri, and Andrew McGregor. Catching the Head, Tail, and Everything in Between: A Streaming Algorithm for the Degree Distribution. In *2015 IEEE International Conference on Data Mining, ICDM 2015, Atlantic City, NJ, USA, November 14-17, 2015*, pages 979–984, 2015.
- 42 Johan Ugander, Lars Backstrom, and Jon Kleinberg. Subgraph Frequencies: Mapping the Empirical and Extremal Geography of Large Graph Collections. In *Proceedings of the 22Nd International Conference on World Wide Web, WWW '13*, pages 1307–1318, Republic and Canton of Geneva, Switzerland, 2013. International World Wide Web Conferences Steering Committee. URL: <http://dl.acm.org/citation.cfm?id=2488388.2488502>.
- 43 Yuichi Yoshida, Masaki Yamamoto, and Hiro Ito. An improved constant-time approximation algorithm for maximum matchings. In *Proceedings of the 41st Annual ACM Symposium on Theory of Computing, STOC 2009, Bethesda, MD, USA, May 31 - June 2, 2009*, pages 225–234, 2009.

# Tensor Network Complexity of Multilinear Maps

**Per Austrin**

School of Computer Science and Communication, KTH Royal Institute of Technology,  
Stockholm, Sweden  
austrin@kth.se

**Petteri Kaski**

Department of Computer Science, Aalto University, Helsinki, Finland  
petteri.kaski@aalto.fi

**Kaie Kubjas**

Department of Mathematics and Systems Analysis, Aalto University, Helsinki, Finland, and  
Laboratoire d'Informatique de Paris 6, Sorbonne Université, Paris, France  
kaie.kubjas@aalto.fi

---

## Abstract

We study *tensor networks* as a model of arithmetic computation for evaluating multilinear maps. These capture any algorithm based on low border rank tensor decompositions, such as  $O(n^{\omega+\epsilon})$  time matrix multiplication, and in addition many other algorithms such as  $O(n \log n)$  time discrete Fourier transform and  $O^*(2^n)$  time for computing the permanent of a matrix. However tensor networks sometimes yield faster algorithms than those that follow from low-rank decompositions. For instance the fastest known  $O(n^{(\omega+\epsilon)t})$  time algorithms for counting  $3t$ -cliques can be implemented with tensor networks, even though the underlying tensor has border rank  $n^{3t}$  for all  $t \geq 2$ . For counting homomorphisms of a general pattern graph  $P$  into a host graph on  $n$  vertices we obtain an upper bound of  $O(n^{(\omega+\epsilon) \text{bw}(P)/2})$  where  $\text{bw}(P)$  is the branchwidth of  $P$ . This essentially matches the bound for counting cliques, and yields small improvements over previous algorithms for many choices of  $P$ .

While powerful, the model still has limitations, and we are able to show a number of unconditional lower bounds for various multilinear maps, including:

- (a) an  $\Omega(n^{\text{bw}(P)})$  time lower bound for counting homomorphisms from  $P$  to an  $n$ -vertex graph, matching the upper bound if  $\omega = 2$ . In particular for  $P$  a  $v$ -clique this yields an  $\Omega(n^{\lceil 2v/3 \rceil})$  time lower bound for counting  $v$ -cliques, and for  $P$  a  $k$ -uniform  $v$ -hyperclique we obtain an  $\Omega(n^v)$  time lower bound for  $k \geq 3$ , ruling out tensor networks as an approach to obtaining non-trivial algorithms for hyperclique counting and the Max-3-CSP problem.
- (b) an  $\Omega(2^{0.918n})$  time lower bound for the permanent of an  $n \times n$  matrix.

**2012 ACM Subject Classification** Theory of computation  $\rightarrow$  Models of computation, Theory of computation  $\rightarrow$  Computational complexity and cryptography, Theory of computation  $\rightarrow$  Design and analysis of algorithms

**Keywords and phrases** arithmetic complexity, lower bound, multilinear map, tensor network

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2019.7

**Related Version** A full version of this paper appears at <https://arxiv.org/abs/1712.09630>.

**Funding** Per Austrin was funded by the Swedish Research Council, Grant 621-2012-4546. Petteri Kaski has received funding from the European Research Council, Grant 338077. Kaie Kubjas was supported by Marie Skłodowska-Curie Grant 748354.

**Acknowledgements** We are grateful to Andreas Björklund for highlighting branchwidth to us as a natural parameter to generalize from clique-counting to counting homomorphisms.



© Per Austrin, Petteri Kaski, and Kaie Kubjas;  
licensed under Creative Commons License CC-BY  
10th Innovations in Theoretical Computer Science (ITCS 2019).  
Editor: Avrim Blum; Article No. 7; pp. 7:1–7:21



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

## 1 Introduction

One of the cornerstones of the theory of computation is the study of efficient algorithms:

For a function  $f$ , how much time is required to evaluate  $f$  on given inputs?

Answering this question for almost any specific  $f$  is well beyond reach of contemporary tools. For example, it is theoretically possible that canonical NP-complete problems, such as the Circuit Satisfiability problem, can be solved in linear time whereas they are widely believed to require super-polynomial (or somewhat less widely, exponential) time [34, 35, 36]. The main reason for this barrier to quantitative understanding is that it is very hard to prove lower bounds for explicit functions in general models of computation such as circuits or Turing machines. This situation withstanding, a more modest program to advance our understanding of computation is to study restricted models that for many  $f$  are simultaneously

- (i) general enough to capture the fastest-known algorithms for  $f$ , and
- (ii) restricted enough to admit proofs of strong *unconditional* time lower bounds for  $f$ .

There is a substantial body of work that fits under this program, ranging from the study of low-depth or otherwise restricted circuits (see e.g. [7], Ch. 14) to models of algorithm-design principles such as greedy algorithms, backtracking, or dynamic programming [3, 27], to linear or semidefinite programming relaxations for hard optimization problems [51].

**Multilinear maps.** One class of functions  $f$  that are of substantial importance is the family of  $\ell$ -linear maps (*multilinear maps*) from  $\ell$  input vector spaces to an output vector space.<sup>1</sup> Examples range from maps of known near-linear-time complexity in the input size, such as the discrete Fourier transform [24, 72], to maps without known polynomial-time-complexity algorithms, such as the permanent of a matrix [64, 70]. Beyond motivation in numerical multilinear algebra and its applications, recent advances in the study of fine-grained algorithm design and complexity have highlighted the fundamental role of algebraic methods in the fastest-known algorithm designs across a broad range of tasks from graph problems, such as all-pairs shortest paths and  $k$ -clique, to parsing and constraint satisfaction problems, such as maximum satisfiability and graph coloring [2, 11, 13, 30, 37, 54, 75, 76].

In this paper, we study the *arithmetic complexity* of evaluating a multilinear map, that is, the number of operations (scalar additions, subtractions, and multiplications) needed to evaluate the map. To set up a baseline, a generic  $\ell$ -linear map from  $\ell$  vector spaces of dimension  $n$  to a scalar requires  $\Omega(n^\ell)$  scalars to represent the map directly using combinations of basis vectors. Given this complexity of a direct explicit representation, it is a fundamental problem to seek less costly representations, along with associated efficient algorithms that work on the chosen representation.

We propose the systematic study of *tensor networks* on hypergraphs as a framework for fast evaluation of multilinear maps, and show a number of upper and lower bounds on the computational power of tensor networks in the spirit of (i) and (ii) above.

**Tensor networks.** Tensor networks have a long and rich history which can be traced as far back as 19<sup>th</sup>-century studies in invariant theory due to Cayley [20, 21], Clebsch [22], Clifford [23], Sylvester [69], and Kempe [40, 41]. Tensor networks are extensively deployed in applications from pure mathematics and theoretical physics [39, 47, 48, 49, 57, 58, 61, 65] to computational physics and chemistry [56, 59, 67]. In theoretical computer science, they appear in various guises including, for example, in the Holant framework [71, 18, 17], in the study of

<sup>1</sup> Multilinear maps with  $\ell = 1$  are called *linear*,  $\ell = 2$  *bilinear*,  $\ell = 3$  *trilinear*, and so forth.

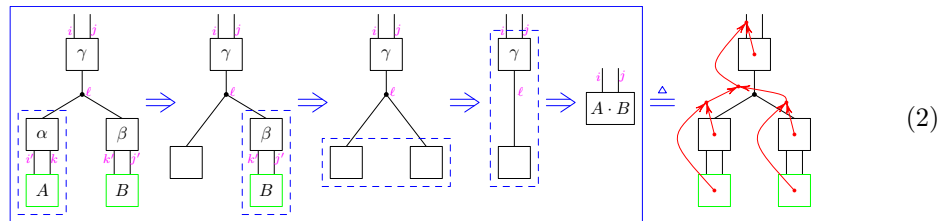
probabilistic graphical models [45, 62], in the study of parallel programming [66], in the study of quantum computing [6], and in the study of arithmetic complexity [8, 19, 26]. Tensor contraction is also emerging as a basic computational primitive in computer hardware [31, 53]. (We refer to the full version of this paper a more detailed discussion.) As the precise definitions are somewhat technical, let us start with a few simple motivating examples and then state our results, with the understanding that precise definitions appear in Section 3.

In our setting, a tensor network is a hypergraph in which the vertices are tensors and the hyperedges are called *modes*. Each mode that is incident to a tensor defines a “dimension” for indexing the entries of the tensor – for example, a matrix is a tensor that is incident to two modes, one mode for the rows of the matrix, and the other mode for the columns of the matrix. A network may be simplified by a sequence of *contractions*, where each contraction takes a subset of tensors and replaces it with a single tensor whose entries are obtained as generalized inner products of the entries of the tensors being contracted.

As a first example of these concepts, let us consider the task of multiplying two matrices,  $A$  and  $B$ . More specifically, let  $A$  be a matrix with rows indexed by mode  $i$  and columns indexed by mode  $k$ , and let  $B$  be a matrix with rows indexed by mode  $k$  and columns indexed by mode  $j$ . We may represent the multiplication task as the tensor network depicted on the left in (1). The result of contracting  $A$  and  $B$  is a new matrix with rows indexed by  $i$  and columns indexed by  $j$ , where the entry at each position  $(i, j)$  is  $\sum_k A_{ik}B_{kj}$ . If the three index sets all have size  $n$ , then computing  $A \cdot B$  by contracting them in such a direct manner uses  $\Theta(n^3)$  operations. To obtain faster matrix multiplication, we can rewrite the bare-bones network on the left in (1) using a low-rank decomposition of the matrix multiplication tensor. For example, Strassen’s decomposition [68] of  $2 \times 2$  matrix multiplication can be represented using the second network in (1). Note that the index  $i$  used by  $A$  and the result has been separated into two distinct indices  $i$  and  $i'$ , and similarly for  $j$  and  $k$ .

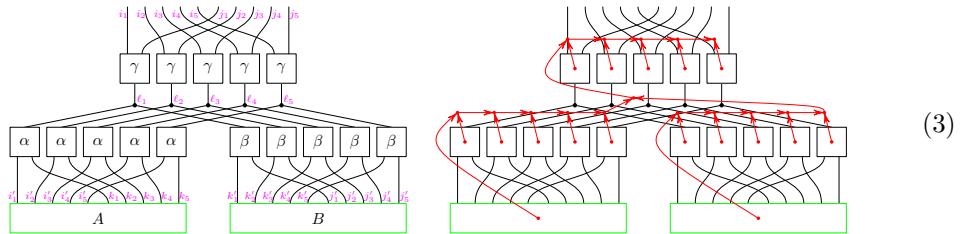
$$\begin{aligned}
 & \text{Network 1: } \begin{array}{c} i \\ \downarrow \\ \boxed{A} \\ \downarrow k \\ \boxed{B} \\ \downarrow j \end{array} \quad \begin{array}{c} i \downarrow j \\ \downarrow \\ \boxed{\gamma} \\ \downarrow e \\ \begin{array}{cc} \boxed{\alpha} & \boxed{\beta} \\ \downarrow i' \downarrow k' & \downarrow k' \downarrow j' \\ \boxed{A} & \boxed{B} \end{array} \end{array} \quad \begin{array}{l} \gamma = \begin{bmatrix} 1 & 0 & 0 & 1 & -1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 & 0 \\ 1 & -1 & 1 & 0 & 0 & 1 & 0 \end{bmatrix} \\ \alpha = \begin{bmatrix} 1 & 0 & 1 & 0 & 1 & -1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & -1 \end{bmatrix} \\ \beta = \begin{bmatrix} 1 & 1 & 0 & -1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & -1 & 0 & 1 & 0 & 1 \end{bmatrix} \end{array} \quad (1)
 \end{aligned}$$

We can *execute* the network by successively contracting groups of vertices. In (2) we see the process of successively contracting pairs of tensors in a carefully chosen order, until only a single tensor – the result of the computation – remains. Such an execution can be naturally represented by a rooted binary tree, as shown on the right in (2), where the tensors of the network form the leaves, and each internal node represents the result of contracting its two children. To summarize, a tensor-network algorithm is specified by providing (a) a tensor network that when contracted yields the desired result, and (b) an execution tree indicating the order in which to contract tensors in the network.



The *cost* of performing one of the contractions in an execution is the product of the lengths of the modes used by any tensor involved in the contraction. This simply measures (up to a constant factor) the number of arithmetic operations (additions/multiplications) used to compute the result by a direct, naïve computation that does not depend on the values of the tensors. For example, the contraction of  $\alpha$  and  $A$  in the first step of (2) has cost 28 because it involves the three modes  $i'$  (length 2),  $k$  (length 2) and  $\ell$  (length 7).

We observe that cost is data-oblivious – the tensor  $\alpha$  is fixed with many zero-entries but these entries still contribute to the cost. Indeed, in many cases there may be faster ways of evaluating a contraction than to evaluate it naively, and just like we saw above, this can often be dealt with by rewriting the network appropriately. For instance, consider now the multiplication of two  $2^k \times 2^k$  matrices. Because the family of matrix multiplication tensors is closed under Kronecker products, this operation may be computed by a tensor network like the one shown in (3) (depicting the case  $k = 5$ ), where  $\alpha$ ,  $\beta$  and  $\gamma$  are as in (2). The rows/columns of the matrices are now indexed by  $k$ -tuples of bits. The execution of this network contracts one  $\alpha/\beta/\gamma$  tensor at a time, which lets us keep the cost low. For example, the first contraction of  $A$  with the first  $\alpha$  block has a cost of  $2^k \cdot 2^k \cdot 7$ , and results in a tensor of size  $2^{k-1} \times 2^{k-1} \times 7$ , then the next contraction has a cost of  $2^{k-1} \cdot 2^{k-1} \cdot 7^2$  and produces a result of size  $2^{k-2} \times 2^{k-2} \times 7 \times 7$ , and so on, until the contraction with the last  $\alpha$  block which has a cost of  $2 \cdot 2 \cdot 7^k = O(7^k)$ , and all the contractions in the execution have cost bounded by this, meaning that we get a total running time of  $O(k7^k) = O(N^{\log_2 7} \log N)$  for  $N \times N$  matrices.<sup>2</sup>



This type of argument can capture any algorithm based on a low-rank decomposition of the underlying tensor of the multilinear map, and indeed, this enables  $O(n^\omega)$ -time<sup>3</sup> matrix multiplication using tensor networks. Beyond simple low-rank decompositions, which always give rise to “star-like” networks as in (3), there are many interesting algorithms that can be captured using networks with a more complicated topology. For instance, many maps of substantial importance have a layered structure that decomposes the map to a sequence of elementary maps. A canonical example is the discrete Fourier transform (DFT), which for a smooth composite order such as  $2^k$ , can be decomposed into a fast Fourier transform (FFT) that consists of a sequence of  $k$  transforms of order 2 interleaved with diagonal-matrix multiplications of twiddle factors [24, 72].

<sup>2</sup> In fact, a more careful analysis gives running time  $O(N^{\log_2 7})$ .  
<sup>3</sup> Throughout the paper,  $\omega = \omega(\mathbb{F})$  denotes the infimum over all  $e$  such that the arithmetic complexity of multiplying two  $n \times n$  matrices is  $O(n^e)$ . While the value of  $\omega$  may depend on the underlying field  $\mathbb{F}$ , we tacitly ignore this, since the field is fixed throughout the paper. For all fields it is known that  $2 \leq \omega < 2.3728639$  [50, 73].



## 1.1 Our results

Starting with motivation (i) and seeking to express existing fastest-known algorithms as executions of tensor networks by a sequence of contractions, we show upper bounds for a number of natural problems. Beyond standard linear settings such as the FFT, not only do tensor networks realize classical bilinear settings such as Abelian group algebra products and fast matrix multiplication algorithms based on low tensor rank, they are powerful enough to capture a substantial number of higher-linearity applications, including Ryser's algorithm for matrix permanent [64], and the *Kruskal operator* [43, 46], which underlies realization of rank-decompositions for tensor rank [44] and current fastest algorithms for detecting outlier correlations [38].

One problem for which tensor networks turn out to be particularly useful is counting homomorphisms from a fixed pattern graph  $P$  to a large host graph  $G$  on  $n$  vertices. The most well-studied such problem is when  $P$  is a  $k$ -clique. For this problem, the currently fastest algorithm runs in time roughly  $O(n^{\omega^{k/3}})$  (with variations in the exponent depending on  $k \bmod 3$ ) [54, 30]. For general  $P$ , it is known that the problem can be solved in  $O(n^{\text{tw}(P)+1})$  time [28], where  $\text{tw}(P)$  is the treewidth of  $P$ . We show that tensor networks can solve the problem in  $O(n^{(\omega+\epsilon)\text{bw}(P)/2})$  time, where  $\text{bw}(P)$  is the *branchwidth* of  $P$ . For  $P$  a  $k$ -clique we have  $\text{bw}(P) = \lceil 2k/3 \rceil$  so this almost recovers the  $O(n^{\omega^{k/3}})$  running time, and in this case we can slightly improve the running time to recover the  $O(n^{\omega^{\lfloor k/3 \rfloor + (k \bmod 3)}}$ ) time of Nešetřil and Poljak [54]. In the case of general  $P$ , this improves on the treewidth-based bound for graphs with  $\text{bw}(P) \leq 2(\text{tw}(P) + 1)/\omega$  (and in particular if  $\omega = 2$  it is always as fast as the treewidth-based bound, ignoring the  $\epsilon$ ). By recent results of Curticapean, Dell, and Marx [25], fast algorithms for homomorphism-counting can be used to obtain fast algorithms for counting subgraphs of  $G$  isomorphic to  $P$ , and in some cases our new branchwidth-based bound leads to an improvement; for example, for counting paths of lengths of length 7, 8 or 9, we get a running time of  $O(n^{3\omega/2+\epsilon}) < O(n^{3.56})$  compared to  $O(n^4)$  using the treewidth-based bound, whereas for very long paths it is not clear whether we would need  $\omega = 2$  in order for these bound to improve on the treewidth-based bound. Previous work that combines branch decompositions and fast matrix multiplication includes Dorn [29] and Bodlaender *et al.* [15].

Further applications captured by tensor networks are the set covering and set partitioning frameworks via fast zeta and Möbius transforms that underlie the current fastest algorithms for graph coloring [13] and its generalizations such as computing the Tutte polynomial [10, 11]. To summarize, we have the following compendium theorem of upper bound results.

► **Theorem 1.1.** *We have the following upper bounds on arithmetic complexity via tensor networks:*

1.  $O(n^{\omega+\epsilon})$  for the matrix multiplication map of two  $n \times n$  matrices.
2.  $O(n^{(\omega+\epsilon)\lfloor v/3 \rfloor + (v \bmod 3)})$  for counting  $v$ -cliques in an  $n$ -vertex graph.
3.  $O(n^{(\omega+\epsilon)\text{bw}(P)/2})$  for counting homomorphisms of a fixed pattern (hyper)graph  $P$  into a (hyper)graph on  $n$  vertices.
4.  $O(\max(n^{\lceil \ell/2 \rceil(\omega+\epsilon-1)r}, n^{2\lceil \ell/2 \rceil r^{\omega+\epsilon-2}}))$  for the Kruskal operator of  $\ell$  matrices of shape  $n \times r$ .
5.  $O(2^k k)$  for the discrete Fourier transforms for the Abelian groups  $\mathbb{Z}_{2^k}$  and  $\mathbb{Z}_2^k$ .
6.  $O(2^k k)$  for group algebra products on  $\mathbb{F}[\mathbb{Z}_{2^k}]$  and  $\mathbb{F}[\mathbb{Z}_2^k]$  when 2 is unit in  $\mathbb{F}$ .
7.  $O(2^k k)$  for the semigroup algebra product on  $\mathbb{F}[\{0, 1\}^k, \subseteq, \cap, \cup]$ .
8.  $O(2^n n)$  for the permanent of an  $n \times n$  matrix.

Above  $\epsilon > 0$  is an arbitrary constant.

Perhaps the most interesting application above is the  $v$ -clique problem, which suggests that one should seek to pursue generalizations to  $v$ -vertex hypercliques of  $\binom{v}{k}$  hyperedges with  $v > k \geq 3$ . Indeed, subgraph counting is a problem that has received substantial interest over the years (e.g. [37, 54, 5, 4, 30, 12, 14, 77, 74, 33, 32, 55, 42, 25]), but progress in the particular case of  $v$ -clique has been stuck to the extent that the problem has attracted notoriety as a hardness assumption in fine-grained complexity [1, 2]. Beyond the study of cliques, hypercliques, and subgraph counting, nontrivial algorithms for such forms would have immediate applicability, for example, in the study of maximum constraint satisfaction problems (Max-CSP) for constraints of width  $k \geq 3$ ; cf. Williams [75] for the case  $k = 2$ . One of the main goals of our subsequent lower bounds is to rule out tensor networks as a candidate to yield improved algorithms in this setting.

Turning from upper bounds to lower bounds and motivation (ii), tensor networks are restricted enough to enable nontrivial lower bounds for many multilinear maps. To begin with, an immediate limitation of tensor networks is that all the intermediate results during the execution of a network are multilinear, and the execution of a network can be simulated by a multilinear circuit. Raz [60] shows that multilinear formulas cannot compute the determinant of an  $n \times n$  matrix in a polynomial number of operations in  $n$ , even though polynomial-size general circuits are known for the determinant (cf. [9, 16, 63]).

It turns out that considerably stronger lower bounds can be shown for tensor networks. In particular, we establish essentially tight lower bounds (subject to the assumption  $\omega = 2$ ) for arithmetic complexity via tensor networks of  $P$ -homomorphism counting and the Kruskal operator. Furthermore, we rule out the possibility of any nontrivial algorithm designs via tensor networks for counting cliques in hypergraphs. The following theorem collects our main lower-bound results, and should be contrasted with the upper bounds in Theorem 1.1.

► **Theorem 1.2.** *We have the following lower bounds on arithmetic complexity via tensor networks:*

1.  $\Omega(n^{\text{bw}(P)})$  for the multilinear form corresponding to  $P$ -homomorphism counting. In particular, this yields a lower bound of  $\Omega(n^{\lceil 2v/3 \rceil})$  for counting cliques of size  $v$ , and a lower bound of  $\Omega(n^v)$  for counting hypercliques of size  $v$ .
2.  $\Omega(\max(n^\ell, n^{\lceil \ell/2 \rceil} r))$  for the Kruskal operator of  $\ell$  matrices of shape  $n \times r$ .
3.  $\Omega(\binom{n}{\lfloor n/3 \rfloor})$  for the determinant or permanent of an  $n \times n$  matrix.

We remark that [52] independently showed that the border rank of the  $v$ -hyperclique tensor is  $\Omega(n^v)$ ; our  $\Omega(n^v)$  lower bound for tensor networks strengthens that. One may wonder about the gap between the bounds of  $\Omega(\binom{n}{\lfloor n/3 \rfloor})$  and  $O(2^n n)$  for the permanent. As we explain below, our lower bound methods are inherently rank-based and cannot go beyond  $\binom{n}{\lfloor n/3 \rfloor}$ . A curious point is that it is not immediately clear whether tensor networks can even achieve  $O^*(2^n)$  time for the determinant, and we do not know whether or not this is possible.

## 1.2 Overview of proof ideas

As a running example in this overview, we consider the 6-linear forms  $A : \mathbb{F}^{[n] \times [n]} \times \mathbb{F}^{[n] \times [n]} \times \dots \times \mathbb{F}^{[n] \times [n]} \rightarrow \mathbb{F}$  taking as input 6 matrices of size  $n \times n$ , defined by

$$A(X^{(1)}, X^{(2)}, X^{(3)}, X^{(4)}, X^{(5)}, X^{(6)}) = \sum_{i,j,k,\ell \in [n]} X_{ij}^{(1)} X_{ik}^{(2)} X_{il}^{(3)} X_{jk}^{(4)} X_{j\ell}^{(5)} X_{k\ell}^{(6)}. \quad (4)$$

If  $\chi$  is the adjacency matrix of a loopless graph  $G$ , then  $A(\chi, \chi, \chi, \chi, \chi, \chi)$  counts the number of 4-cliques in the graph. Associated with  $A$  is the 6-tensor  $T(A)$  of size  $n^2 \times n^2 \times \dots \times n^2$ ,



where each of the 6 modes is indexed by a pair  $(i, j) \in [n] \times [n]$ , and the value at a given position is the coefficient of the corresponding term in  $A$ . Concretely,

$$T(A)_{i_1 j_1, i_2 k_2, i_3 \ell_3, j_4 k_4, j_5 \ell_5, k_6 \ell_6} = \begin{cases} 1 & \text{if } i_1 = i_2 = i_3, j_1 = j_4 = j_5, k_2 = k_4 = k_6 \wedge \ell_3 = \ell_5 = \ell_6, \\ 0 & \text{otherwise.} \end{cases}$$

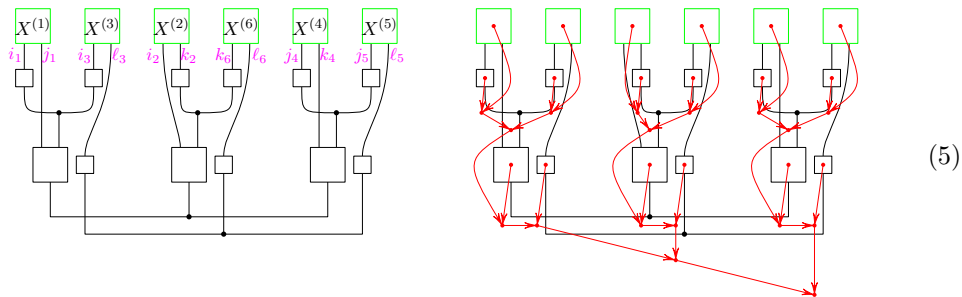
**Upper bounds.** Most, but not all, of the families of multilinear maps we consider are closed under taking Kronecker products. For instance, consider the 4-clique counting form (4) for an  $n$ -vertex graph and its associated tensor  $T(A)$ . Then for any  $k \geq 1$ , the tensor associated with the 4-clique counting form in  $n^k$ -vertex graphs is  $T(A)^{\otimes k}$ , the  $k$ -fold Kronecker product of  $T(A)$  with itself. We write  $A^{\otimes k}$  for the associated map. With this in mind, it is natural to seek general constructions that, given an efficient evaluation of some map  $A$ , yields an efficient evaluation of  $A^{\otimes k}$ .

We give such a construction, and show that the cost of the best tensor network execution for  $A^{\otimes k}$  is essentially submultiplicative in a quantity that we call the *amortized cost* of an execution. For tensors of order at most 3, the notion of amortized cost essentially captures the rank of  $T(A)$ , but for higher-order tensors, the amortized cost may be significantly smaller than the rank. Roughly speaking, the amortized cost of a step in an execution of a map  $A$  is: (i) equal to the normal cost if the operation involves the contraction of two tensors that both depend on some input variables of  $A$ , but (ii) equal to the size of the result if only one of the tensors involved in the contraction depends on the input variables of  $A$ . A precise definition appears in Section 4. Our general upper bound for the cost of  $A^{\otimes k}$  can, somewhat informally, be stated as follows.

► **Theorem 1.3** (Submultiplicativity of cost, informal statement). *If a multilinear map  $A$  has a tensor network execution consisting of  $s$  steps, each with cost at most  $c$  and amortized cost at most  $a$ , then  $A^{\otimes k}$  has a tensor network execution consisting of at most  $k \cdot s$  steps, each with cost at most  $a^{k-1} \cdot c$ .*

An immediate corollary of this is that we can capture any algorithm for  $A^{\otimes k}$  based on a low-rank decomposition of  $T(A)$  (Corollary 4.2). For example, this implies that tensor networks can multiply  $n \times n$  matrices in  $O(n^{\omega+\epsilon})$  time.

However, returning to our running example form (4), as we explain below the tensor  $T(A)$  has rank  $n^4$ , meaning that Corollary 4.2 only yields a trivial upper bound. This is where the full generality of Theorem 1.3 comes in. Consider the form (4) for graphs on some constant number  $n_0$  of vertices. As it turns out, we can design a network and an associated execution for this form, depicted in (5) and explained in more detail in the full version of this paper, with an execution of cost  $n_0^{2e+3}$  and amortized cost  $n_0^{e+1}$ , where  $n_0^e$  is the rank of the tensor associated with  $n_0 \times n_0$  matrix multiplication. Picking  $n_0$  to be a large enough constant so that  $e$  is approximately  $\omega$ , and letting  $k$  be such that  $n$  is approximately  $n_0^k$ , we obtain via Theorem 1.3 an  $O(n^{\omega+\epsilon+1})$  time upper bound for (4).



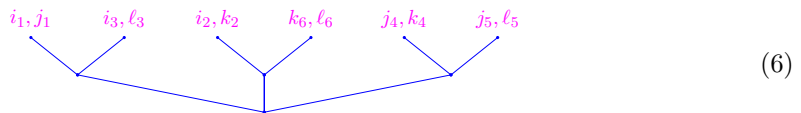
**Lower bounds.** Just like many other arithmetic complexity lower bounds, our lower bounds boil down to establishing lower bounds on the rank of certain matrices.

To establish a lower bound on the *rank* of  $T(A)$ , we *flatten* it to a matrix and analyze the rank of that matrix. There are  $2^5$  possible bipartitions of the set of 6 modes of  $T(A)$ , and the lower bound on the rank of  $T(A)$  that we obtain is the maximum of the ranks of the resulting matrices. Using this method it is easy to establish that for our example form (4), the rank of  $T(A) = n^4$ . That this is an upper bound follows from (4), and that it is a lower bound follows by considering the bipartition taking variables  $X^{(1)}$  and  $X^{(6)}$  as row indices, and the other 4 variables as column indices. The resulting  $n^4 \times n^8$  matrix has full rank.

Tensor networks are more versatile and can be more efficient than low-rank decompositions of  $T(A)$ . Nevertheless, we show limitations on this versatility. In particular we show that every tensor network execution for  $A$  induces a tree in which the leaves are the inputs of  $A$  and all internal vertices have degree 3. We call this a *socket tree*. Each edge in a socket tree induces a bipartition of the variables and our key technical lemma is to show that for each such bipartition, the rank of the corresponding flattening of  $T(A)$  is a lower bound on the cost of the execution that gave rise to the tree. Thus, to obtain a lower bound for the cost of a specific execution, we consider the maximum rank obtained over all edges of the corresponding socket tree, and to lower bound the cost of every tensor network execution, we minimize this quantity over all possible socket trees. We refer to the resulting quantity as the *socket width* of  $A$ , denoted  $w(A)$  (formal definition appears in Section 5). Our general lower bound can thus be phrased as follows, where  $c(A)$  denotes the minimum cost of a tensor network for evaluating  $A$  (formal definition appears in Section 3).

► **Theorem 1.4.** *For every multilinear map  $A$ , it holds that  $c(A) \geq w(A)$ .*

Indeed, for our running example (4), there are low-width socket trees establishing that  $w(A) \leq n^3$ , see (6). However, that bound is tight: our  $\Omega(n^{\lceil 2 \cdot 4/3 \rceil}) = \Omega(n^3)$  lower bound for the  $\binom{4}{2}$ -linear form (Theorem 1.2) is obtained by proving that  $w(A) \geq n^3$  and appealing to Theorem 1.4.



### 1.3 Organization of this paper

The present conference abstract contains only the key definitions and technical results underlying our main theorems. All the upper and lower bounds for the arithmetic complexity of specific multilinear maps together with their proofs can be found in the full version of this paper. Section 2 recalls preliminaries on tensors and multilinear maps. In Section 3, tensor networks, execution and cost of a tensor network, and cost of a multilinear map are defined. Section 4 presents our main upper-bound result on submultiplicativity of cost. In Section 5, a general lower bound on the cost of evaluating a multilinear map using tensor network is obtained; this lower bound is expressed in terms of the socket-width of a multilinear map.

## 2 Preliminaries on tensors and multilinear maps

This section sets up our notation for tensors and multilinear maps. Throughout the paper  $[n]$  denotes  $\{1, 2, \dots, n\}$  and  $\mathbb{F}$  denotes an arbitrary fixed field. We work with tensors and multilinear maps relative to fixed bases for the respective vector spaces over  $\mathbb{F}$ .

**Modes, indexing, and positions.** We will work with the following convention of positioning individual entries inside a tensor. Let  $E$  be a finite set of *modes*. Associate with each mode  $e \in E$  a finite nonempty *index set*  $J(e)$ . In this case we say that  $E$  is a set of *indexed modes*. The *length* of  $e$  is  $|J(e)|$ . A *position* is an element  $j \in \prod_{e \in E} J(e)$ . Let us write  $J(E) = \prod_{e \in E} J(e)$  for the set of all positions with respect to the indexed modes  $E$ . In the special case that the set of modes  $E$  is empty we define the set of positions  $J(E)$  to consist of a single element.

**Tensors, matrices, vectors, and scalars.** Let  $E$  be a set of indexed modes. A *tensor* with respect to  $E$  is a mapping  $T : J(E) \rightarrow \mathbb{F}$ . Equivalently, we write  $T \in \mathbb{F}^{J(E)}$  to indicate that  $T$  is a tensor with respect to the indexed modes  $E$ . We view the set  $\mathbb{F}^{J(E)}$  of all tensors over  $E$  as a vector space over  $\mathbb{F}$  with addition and scalar multiplication of tensors defined entrywise. We say that  $|E|$  is the *order* of the tensor. A tensor of order zero is called a *scalar*, a tensor of order one is called a *vector*, and a tensor of order two is called a *matrix*. The *volume* of the tensor is  $|J(E)|$ . The tuple  $(|J(e)| : e \in E)$  is the *shape* of the tensor. It is convenient to use the “ $\times$ ”-symbol to punctuate the shape of a tensor; that is, instead of writing, say  $(2, 3, 4)$  for the shape, we write  $2 \times 3 \times 4$ . For a position  $j \in J(E)$  and a tensor  $T \in \mathbb{F}^{J(E)}$ , we say that  $T_j \in \mathbb{F}$  is the *entry* of  $T$  at  $j$ .

A *flattening* of  $T$  induced by a bipartition  $E_1 \cup E_2 = E$  of the modes of  $T$  is a  $|J(E_1)| \times |J(E_2)|$  matrix  $M$  where, for  $j_1 \in J(E_1)$  and  $j_2 \in J(E_2)$  we have  $M_{j_1, j_2} = T_{j_1, j_2}$ . Given two order  $\ell$  tensors  $S \in \mathbb{F}^{[n_1] \times [n_2] \times \dots \times [n_\ell]}$  and  $T \in \mathbb{F}^{[m_1] \times [m_2] \times \dots \times [m_\ell]}$ , their *Kronecker product*  $S \otimes T$  is a tensor in  $\mathbb{F}^{[n_1 m_1] \times [n_2 m_2] \times \dots \times [n_\ell m_\ell]}$  defined by

$$(S \otimes T)_{m_1(i_1-1)+j_1, m_2(i_2-1)+j_2, \dots, m_\ell(i_\ell-1)+j_\ell} = S_{i_1, i_2, \dots, i_\ell} T_{j_1, j_2, \dots, j_\ell}.$$

**Multilinear maps.** Let  $E_1, E_2, \dots, E_\ell, E'$  be pairwise disjoint sets of indexed modes such that  $E_1, E_2, \dots, E_\ell$  are nonempty. A map  $A : \mathbb{F}^{J(E_1)} \times \mathbb{F}^{J(E_2)} \times \dots \times \mathbb{F}^{J(E_\ell)} \rightarrow \mathbb{F}^{J(E')}$  is an  $\ell$ -*linear map* if  $A$  is linear with respect to each of its  $\ell$  inputs individually. In particular, a 1-linear map is a linear map. A multilinear map that gives a scalar output is a *multilinear form*. In particular,  $A$  is a form if and only if  $E'$  is empty.

**The tensors of a multilinear map.** For an  $\ell$ -linear map  $A : \mathbb{F}^{J(E_1)} \times \mathbb{F}^{J(E_2)} \times \dots \times \mathbb{F}^{J(E_\ell)} \rightarrow \mathbb{F}^{J(E')}$ , we define two slightly different tensors  $T(A)$  and  $\hat{T}(A)$ . Both are indexed by  $J(E_1 \cup E_2 \cup \dots \cup E_\ell \cup E')$  and at position  $j_1 j_2 \dots j_\ell j'$  take the value

$$T(A)_{j_1 j_2 \dots j_\ell j'} = \hat{T}(A)_{j_1 j_2 \dots j_\ell j'} = A(e^{(j_1)}, e^{(j_2)}, \dots, e^{(j_\ell)})_{j'},$$

where  $e^{(j_i)} \in \mathbb{F}^{J(E_i)}$  denotes the tensor with a 1 in position  $j_i$  and 0s in all other position. The difference between  $T(A)$  and  $\hat{T}(A)$  is their shape. The shape of  $T(A)$  is  $|J(E_1)| \times |J(E_2)| \times \dots \times |J(E_\ell)| \times |J(E')|$ , except if  $A$  is a form in which case the  $|J(E')|$  part is omitted. Thus  $T(A)$  is of order  $\ell + 1$  (or  $\ell$  if  $A$  is a form). The shape of  $\hat{T}(A)$  is  $(|J(e)| : e \in E_i, i \in [\ell + 1])$ , thus its order is  $|E_1| + |E_2| + \dots + |E_\ell| + |E'|$ .

## 7:10 Tensor Network Complexity of Multilinear Maps

In other words, each mode of  $T(A)$  corresponds to one of the inputs of  $A$ , or the output. These inputs are in turn sets of indexed modes so may contain more “fine-grained” structure, but this information is lost at the level of granularity of  $T(A)$ . When working with tensor networks for evaluating  $A$ , we need to keep track of the fine-grained mode structure because this is in many cases what allows us to construct efficient algorithms, hence in most parts of the paper we are more interested in the tensor  $\hat{T}(A)$  which contains this structure.

On the other hand,  $\hat{T}(A)$  does not contain information about which modes are grouped together to form the inputs and output of  $A$ , and this information is also important. This leads us to the notion of sockets, defined next.

**Sockets.** Let us study the tensor  $\hat{T}(A)$  with respect to the map  $A$ . We say that the modes in  $E_1 \cup E_2 \cup \dots \cup E_\ell$  are the *input* modes of  $\hat{T}(A)$ , and the modes in  $E'$  are the *output* modes of  $\hat{T}(A)$  with respect to  $A$ . Let us say that  $E_1, \dots, E_\ell$  are the *input sockets* of  $\hat{T}(A)$  with respect to  $A$ . Similarly,  $E'$  is the *output socket* in  $\hat{T}(A)$  with respect to  $A$ . In particular, the output socket is empty if and only if  $A$  is a form. To describe a socketing of the modes of a tensor, it is convenient to use parentheses to group a “ $\times$ ”-punctuated shape of a tensor into sockets, see also Section 2.

Let  $\hat{T}$  be a tensor over a set of indexed modes  $E$ . Any tuple  $\mathcal{E} = (E_1, E_2, \dots, E_\ell, E')$  of subsets of  $E$  that partitions  $E$  with  $E_1, E_2, \dots, E_\ell$  nonempty defines an  $\ell$ -linear map  $A_{\mathcal{E}}(\hat{T}) : \mathbb{F}^{J(E_1)} \times \mathbb{F}^{J(E_2)} \times \dots \times \mathbb{F}^{J(E_\ell)} \rightarrow \mathbb{F}^{J(E')}$  with  $\hat{T}(A_{\mathcal{E}}(\hat{T})) = \hat{T}$ . In this case the tuple  $(E_1, E_2, \dots, E_\ell)$  gives the input sockets of  $T$  and  $E'$  is the output socket of  $\hat{T}$  with respect to  $A_{\mathcal{E}}(\hat{T})$ . We thus conclude that two multilinear maps  $A_1, A_2$  may have the same base tensor  $\hat{T}(A_1) = \hat{T}(A_2)$ , and from a tensor  $\hat{T}$  one may obtain different multilinear maps by varying how the modes of  $\hat{T}$  are assigned to input and output sockets.

**The form of a multilinear map.** Let  $A$  be a multilinear map with a nonempty output socket. We can turn  $A$  into a multilinear form  $F(A)$  by turning its output socket into an input socket. Let us say that  $F(A)$  is the *multilinear form* of  $A$ . We also set  $F(A) = A$  when  $A$  is a multilinear form.

### 3 Tensor networks

This section defines tensor networks and the cost of a multilinear map.

**Networks.** A *network* (or *diagram*) consists of a finite set  $V$  of *vertices*, a finite set  $E$  of *hyperedges*, an *incidence relation*  $I \subseteq V \times E$ , and a *boundary*  $B \subseteq E$ . A network is *nondegenerate* if every hyperedge is incident to at least one vertex. In what follows we assume that all networks are nondegenerate. A hyperedge  $e \in E$  is a *loop* if  $e \notin B$  and  $e$  is incident to exactly one vertex.

For a vertex  $v \in V$ , let us write  $I(v) = \{e \in E : (v, e) \in I\}$  for the set of hyperedges incident to  $v$ . Dually, for an hyperedge  $e \in E$ , let us write  $I(e) = \{v \in V : (v, e) \in I\}$  for the set of vertices incident to  $e$ . For a network  $D$ , we write  $V(D)$ ,  $E(D)$ ,  $I(D)$ , and  $B(D)$  to refer to the vertices of  $D$ , the hyperedges of  $D$ , the incidence relation of  $D$ , and the boundary of  $D$ , respectively.

**Induced networks.** For a network  $D$  and a nonempty subset  $W \subseteq V(D)$ , the *induced* network  $D[W]$  consists of the vertices in  $W$  together with the hyperedges of  $D$  that are

incident to at least one vertex in  $W$ ; the boundary of  $D[W]$  consists of all hyperedges that are at the boundary of  $D$  or incident to a vertex outside  $W$ . Formally,

$$\begin{aligned} V(D[W]) &= W, \\ E(D[W]) &= \{e \in E(D) : \exists w \in W \text{ s.t. } (w, e) \in I(D)\}, \\ I(D[W]) &= I(D) \cap (V(D[W]) \times E(D[W])), \\ B(D[W]) &= (B(D) \cap E(D[W])) \cup \{e \in E(D[W]) : \exists v \in V(D) \setminus W \text{ s.t. } (v, e) \in I(D)\}. \end{aligned} \tag{7}$$

For a vertex  $v \in V$ , we abbreviate  $D[v] = D[\{v\}]$ . Note that the boundary of  $D[v]$  consists of all non-loop hyperedges incident to  $v$  in  $D$ .

**Tensor networks.** Let  $D$  be a network. We *index*  $D$  by associating with each hyperedge  $e \in E$  an *index set*  $J(e)$  of size  $\ell(e)$ . Induced networks inherit indexing by restriction. Next we associate with each vertex  $v \in V$  a tensor  $T(v) \in \mathbb{F}^{J(I(v))}$ . We say that  $D$  equipped with the tensors  $(T(v))_{v \in V}$  is a *tensor network*.

The *value* of a tensor network  $D$ , or the tensor *represented by*  $D$ , is a tensor  $T(D) \in \mathbb{F}^{J(B)}$ , defined for all  $i \in J(B)$  by

$$T(D)_i = \sum_{j \in J(E(D) \setminus B)} \prod_{v \in V} T(v)_{ij}. \tag{8}$$

Observe that in (8) the positions  $i$  and  $j$  together identify a unique entry of  $T(v)$  by projection to  $J(I(v))$ . The value of a tensor network with an empty boundary is a scalar.

**Contracting tensors.** Let  $D$  be a tensor network and let  $W \subseteq V(D)$  be a nonempty set of vertices. Let  $w$  be a new element not in  $V$ . We may *contract*  $W$  in  $D$  to obtain a tensor network  $D/W$  by replacing the sub-network  $D[W]$  in  $D$  with the single vertex  $w$  whose associated tensor  $T(w)$  is the tensor represented by  $D[W]$ . Formally,

$$\begin{aligned} V(D/W) &= (V(D) \setminus W) \cup \{w\}, \\ E(D/W) &= E(D) \setminus (E(D[W]) \setminus B(D[W])), \\ I(D/W) &= (I(D) \setminus I(D[W])) \cup \{(w, e) : e \in B(D[W])\}, \\ B(D/W) &= B(D), \\ T(w) &= T(D[W]). \end{aligned} \tag{9}$$

The *cost* of contracting  $W$  in  $D$  is  $c(D, W) = \prod_{e \in E(D[W])} |J(e)|$ . The value of a tensor network is invariant under contraction, i.e., for all nonempty  $W \subseteq V(D)$  it holds that  $T(D) = T(D/W)$ .

**Execution and cost of a tensor network.** To compute the tensor  $T(D)$  from a given tensor network  $D$ , we may proceed by a sequence of contractions on  $D$ . Such a process is called *executing*  $D$ , and the *cost* of  $D$  is the cost of a minimum-cost execution of  $D$ .

More precisely, let  $D = D_0$  be a tensor network with at least one tensor. For  $k = 1, 2, \dots, t$ , select a nonempty subset  $W_{k-1} \subseteq V(D_{k-1})$  such that  $W_{k-1}$  has at least two tensors or consists of a single tensor with a loop. Set  $D_k = D_{k-1}/W_{k-1}$  and observe that the number of tensors and/or modes decreases by at least one in the contraction. Suppose that  $D_t$  is loopless and consists of a single tensor. We say that such a sequence of contractions is an *execution* of  $D$  in  $t$  *steps*. The *cost* of the execution is  $\max_{k=1,2,\dots,t} c(D_{k-1}, W_{k-1})$ . The cost of an execution in zero steps is defined to be 0.

It is immediate that  $D$  has at least one execution and every execution consists of at most  $2|V(D)| - 1$  steps. By invariance under contractions, we have  $T(D_t) = T(D)$ . The *cost*  $c(D)$  of  $D$  is the cost of a minimum-cost execution of  $D$ .

An execution of  $D$  of cost  $c(D)$  immediately translates into an algorithm that computes  $T(D)$  using  $O(c(D)|V(D)|)$  arithmetic operations in  $\mathbb{F}$ , since the contraction step  $D_k = D_{k-1}/W_{k-1}$  takes  $O(c(D_{k-1}, W_{k-1})) \leq c(D)$  time to evaluate, and there are  $O(V(D))$  steps.

► **Lemma 3.1.** *Let  $D$  be a tensor network. There exists a minimum-cost execution of  $D$  such that each contracted set has size at most two. Furthermore, if  $D$  is loopless, we can assume that each contracted set has size exactly two.*

In what follows we restrict to consider loopless  $D$  only. Thus while a general execution may contract arbitrary vertex sets in  $D$  in each step, without loss of generality the minimum-cost execution has the structure of a rooted binary tree, whose leaves are the vertices of the tensor network, and each internal vertex is the tensor obtained by contracting its two children.

**Cost of a multilinear map.** We now define the cost of a multilinear map via the minimum-cost tensor networks (and socketing) for evaluating the map. That is, the cost of a map is defined in terms of the best tensor network that implements the map. More precisely, let

$$A : \mathbb{F}^{J(E_1)} \times \mathbb{F}^{J(E_2)} \times \dots \times \mathbb{F}^{J(E_\ell)} \rightarrow \mathbb{F}^{J(E')}$$

be an  $\ell$ -linear map. Consider the tensor  $\hat{T}(A)$  of  $A$  and the associated input sockets  $E_1, E_2, \dots, E_\ell$  and the output socket  $E'$ . Let  $D^*$  be an arbitrary tensor network such that  $T(D^*) = \hat{T}(A)$  and the boundary satisfies  $B(D^*) = E_1 \cup E_2 \cup \dots \cup E_\ell \cup E'$ . Modify the network  $D^*$  as follows. For each  $k = 1, 2, \dots, \ell$ , introduce a new vertex to  $D^*$ , make the new vertex incident to each of the modes in the input socket  $E_k$ , and associate the new vertex with a tensor  $X^{(k)} \in \mathbb{F}^{J(E_k)}$ . Remove the modes  $E_1 \cup E_2 \cup \dots \cup E_\ell$  from the boundary of  $D^*$ . Let us denote the resulting network by  $D$  and call the introduced  $\ell$  new vertices the *socket vertices* of  $D$ . We observe that  $B(D) = E'$  and  $A(X^{(1)}, X^{(2)}, \dots, X^{(\ell)}) = T(D)$ . Furthermore, the cost  $c(D)$  is independent of the value of  $X^{(k)} \in \mathbb{F}^{J(E_k)}$  for  $k = 1, 2, \dots, \ell$ . We say that  $D$  is a *realization* of  $A$  if it can be obtained from  $A$  by this process, and write  $\mathcal{D}(A)$  for the set of all tensor network realizations  $D$  of  $A$ .

The *cost* of the map  $A$  is  $c(A) = \min_{D \in \mathcal{D}(A)} c(D)$ . This minimum exists since the cost of a tensor network is a nonnegative integer and the family  $\mathcal{D}(A)$  is nonempty.

#### 4 An upper bound via submultiplicativity

This section presents our main tool for proving upper bounds on the cost of a multilinear map that admits representation as a Kronecker power, namely submultiplicativity of an amortized notion of cost under Kroneckerling.

**Kronecker power of a multilinear map.** Let  $E_1, E_2, \dots, E_\ell, E'$  be pairwise disjoint sets of indexed modes such that  $E_1, E_2, \dots, E_\ell$  are nonempty. Let

$$A : \mathbb{F}^{J(E_1)} \times \mathbb{F}^{J(E_2)} \times \dots \times \mathbb{F}^{J(E_\ell)} \rightarrow \mathbb{F}^{J(E')}$$

be an  $\ell$ -linear map. For a positive integer  $k$ , we define the  $\ell$ -linear map  $A^{\otimes k}$  such that its tensor satisfies  $T(A^{\otimes k}) = T(A)^{\otimes k}$ . Then

$$A^{\otimes k} : \mathbb{F}^{J(E_1)^k} \times \mathbb{F}^{J(E_2)^k} \times \dots \times \mathbb{F}^{J(E_\ell)^k} \rightarrow \mathbb{F}^{J(E')^k}.$$

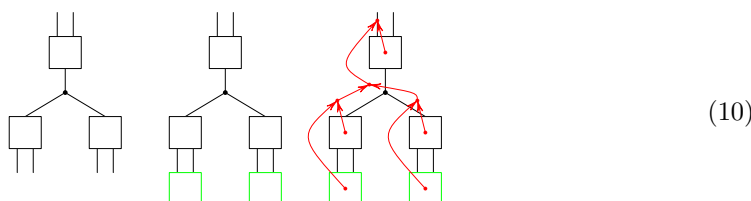
Note that  $T(A^{\otimes k}) = T(A)^{\otimes k}$  is the  $k$ -fold Kronecker product of  $T(A)$  with itself – that is, it has the same order, but the index sets are larger – whereas  $\hat{T}(A^{\otimes k})$  is the  $k$ -fold outer product of  $\hat{T}(A)$  with itself – that is, its index sets have the same sizes, but its order is  $k$  times larger.

**Amortized cost and submultiplicativity.** Let  $D$  be a diagram that realizes  $A$  and let  $\mathcal{T}_D$  be an execution tree for  $D$ . For each internal vertex  $x$  in  $\mathcal{T}_D$  (that is, a vertex obtained by contraction), define the *amortized cost* of  $x$  by splitting into the following three cases:

- (i) if neither of the two subtrees of  $x$  contains a socket vertex, the amortized cost of  $x$  is 1;
- (ii) if exactly one of the subtrees of  $x$ , say, the subtree rooted at  $y$  (where  $x$  and  $y$  are adjacent in  $\mathcal{T}_D$ ), contains at least one socket vertex, the amortized cost of  $x$  is the maximum of the volume of the tensor at  $x$  and the volume of the tensor at  $y$ ;<sup>4</sup>
- (iii) if both of the subtrees of  $x$  contain at least one socket vertex, the amortized cost of  $x$  is the cost of the contraction to obtain  $x$ .

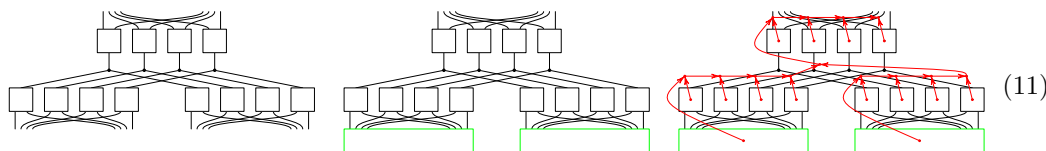
The *amortized cost*  $a(\mathcal{T}_D)$  of  $\mathcal{T}_D$  is the maximum of the amortized costs of the internal vertices of  $\mathcal{T}_D$ . Since the amortized cost of each internal vertex of  $\mathcal{T}_D$  is at most its cost, we have  $a(\mathcal{T}_D) \leq c(\mathcal{T}_D)$ . Furthermore, we observe that the amortized cost of  $x$  in case (ii) above may be strictly less than the cost of the contraction to obtain  $x$ . In particular, in (ii) the amortized cost is defined *not by the cost of the contraction but rather by volume*. This is because in a  $k^{\text{th}}$  Kronecker power we can amortize the cost of the aggregate transformation in case (ii) not with a single contraction but with a sequence of  $k$  contractions. This observation will form the heart of the proof of Theorem 1.3.

Before proceeding with the proof, let us illustrate the key ideas in visual terms. Let us start with the three illustrations in (10).



Suppose the leftmost network in (10) is socketed so that the two modes at the top form the output socket, and the four modes at the bottom form two input sockets so that modes in the same socket are incident to the same vertex. In the middle in (10), we have adjoined two socket vertices to the input sockets to obtain a realization  $D$ . On the right in (10), we display an execution tree  $\mathcal{T}_D$  for  $D$ . Observe that the bottom-most internal vertices of  $\mathcal{T}_D$ , and the top-most internal vertex of  $\mathcal{T}_D$ , have type (ii). The internal vertex in the center has type (iii). (There are no internal vertices of type (i).) Supposing that all the modes have length at least 2, we also observe that the vertices of type (ii) have amortized cost strictly less than their contraction cost.

Let us now consider the  $k^{\text{th}}$  power of (10) visually, for  $k = 4$ :



<sup>4</sup> Here, it is crucial to note that the volume of the other subtree rooted at  $x$ , only containing non-socket vertices, does not contribute directly to the amortized cost of  $x$ .



The leftmost network in (11) depicts the  $k$ -fold outer product of the network on the left in (10) with itself. Observe that we simply take  $k$  copies of the network, but that for the purposes of the visualization we have taken care to draw the  $k$  copies of each mode together for the socketing. In the middle in (11), we have adjoined two socket vertices to the input sockets to obtain a realization  $D^{\otimes k}$  of  $A^{\otimes k}$ . On the right in (11), we display an execution tree  $\mathcal{T}_{D^{\otimes k}}$  for  $D^{\otimes k}$ . Observe how each of the internal vertices of type (ii) in  $\mathcal{T}_D$  gets expanded to a sequence of  $k$  internal vertices in  $\mathcal{T}_{D^{\otimes k}}$ . This transformation from  $\mathcal{T}_D$  to  $\mathcal{T}_{D^{\otimes k}}$  is the gist of the following theorem.

► **Theorem 4.1** (Formal statement of Theorem 1.3). *Let  $D$  be an arbitrary realization of  $A$  and let  $\mathcal{T}_D$  be an arbitrary execution tree for  $D$ . For all positive integers  $k$ , we have*

$$c(A^{\otimes k}) \leq a(\mathcal{T}_D)^{k-1} c(\mathcal{T}_D). \tag{12}$$

Furthermore, this realization of  $A^{\otimes k}$  consists of at most  $k \cdot |V(D)|$  vertices.

**Proof.** Let  $D^*$  be the subnetwork of  $D$  with  $T(D^*) = \hat{T}(A)$ . That is,  $D^*$  is the network induced by all the non-socket vertices of  $D$ . Taking  $k$  disjoint copies of  $D^*$ , we obtain a network whose tensor is  $\hat{T}(A^{\otimes k})$ . Attaching the resulting network to tensors at sockets gives a realization of  $A^{\otimes k}$ . Let us write  $D^{\otimes k}$  for this realization.

To establish (12), it suffices to construct an execution tree  $\mathcal{T}_{D^{\otimes k}}$  for  $D^{\otimes k}$  whose cost satisfies  $c(\mathcal{T}_{D^{\otimes k}}) \leq a(\mathcal{T}_D)^{k-1} c(\mathcal{T}_D)$ . We construct  $\mathcal{T}_{D^{\otimes k}}$  by rewriting  $\mathcal{T}_D$  from leaves towards the root to consider the  $k$  copies of each vertex in  $D^*$ . We start with leaf vertices which are the vertices of  $D^{\otimes k}$ . We split the process into cases (i), (ii), and (iii) as in the definition of amortized cost. Let  $x$  be the internal vertex of  $\mathcal{T}_D$  that we are currently considering.

In case (i), we perform the contraction indicated by  $x$  in each of the  $k$  copies of  $D^*$  in  $D^{\otimes k}$  individually. This creates  $k$  new internal vertices in  $\mathcal{T}_{D^{\otimes k}}$  that are all copies of  $x$ . We set these  $k$  vertices as the vertices that correspond to  $x$  in the subsequent steps. Each of these contractions in  $\mathcal{T}_{D^{\otimes k}}$  has the same cost as the contraction indicated by  $x$  in  $\mathcal{T}_D$ . This cost is less or equal than  $c(\mathcal{T}_D)$ .

In case (ii), let  $y$  be the child of  $x$  in  $\mathcal{T}_D$  such that the subtree rooted at  $y$  contains a socket vertex, and let  $z$  be the other child of  $x$  in  $\mathcal{T}_D$ . There is a single vertex in  $\mathcal{T}_{D^{\otimes k}}$  corresponding to  $y$  and  $k$  identical vertices in  $\mathcal{T}_{D^{\otimes k}}$  corresponding to  $z$ . We contract these  $k$  vertices individually each with the vertex that corresponds to  $y$ . This creates  $k$  new internal vertices in  $\mathcal{T}_{D^{\otimes k}}$ , where we set the topmost vertex as the vertex that corresponds to  $x$  in the subsequent steps. After the  $i$ th step, the corresponding tensor has  $i$  copies of modes of  $x$  and  $k - i$  copies of modes of  $y$ . The cost of the contraction in the  $i$ th step is the cost of contracting  $y$  and  $z$  in  $\mathcal{T}_D$  multiplied by the the volume of  $y$  to the power  $k - i$  and the volume of  $x$  to the power  $i - 1$ . Since the volumes of  $x$  and  $y$  are less than or equal to  $a(\mathcal{T}_D)$ , this cost is less than or equal to  $a(\mathcal{T}_D)^{k-1} c(\mathcal{T}_D)$ .

In case (iii), let  $y$  and  $z$  be the two child vertices of  $x$  in  $\mathcal{T}_D$ . By the structure of the earlier steps, we have that a single vertex in  $D^{\otimes k}$  corresponds to  $y$ , and similarly for  $z$ . We contract these two vertices. This creates one new internal vertex in  $\mathcal{T}_{D^{\otimes k}}$ , which we set as the vertex that corresponds to  $x$  in the subsequent steps. This tensor has  $k$  copies of modes of  $x$ . The cost of this contraction in  $\mathcal{T}_{D^{\otimes k}}$  is the cost of the corresponding contraction in  $\mathcal{T}_D$  to the  $k^{\text{th}}$  power, because both tensors have  $k$  copies of all modes compared to  $y$  and  $z$ . By definition, in case (iii) the amortized cost of contracting  $y$  and  $z$  is the same as the cost of contracting  $y$  and  $z$ . Hence the cost of this contraction in  $\mathcal{T}_{D^{\otimes k}}$  is less or equal than  $a(\mathcal{T}_D)^k \leq a(\mathcal{T}_D)^{k-1} c(\mathcal{T}_D)$ . This rewriting process produces an execution tree  $\mathcal{T}_{D^{\otimes k}}$  for  $D^{\otimes k}$  with  $c(\mathcal{T}_{D^{\otimes k}}) \leq a(\mathcal{T}_D)^{k-1} c(\mathcal{T}_D)$ . ◀



An immediate corollary is that tensor networks can use low rank decompositions of  $T(A)$  to efficiently evaluate  $A^{\otimes k}$ .

► **Corollary 4.2** (Submultiplicativity of low-rank executions). *Let  $A : \mathbb{F}^{J(E_1)} \times \mathbb{F}^{J(E_2)} \times \dots \times \mathbb{F}^{J(E_\ell)} \rightarrow \mathbb{F}^{J(E')}$  be a multilinear map. Define  $m = \max\{|J(E_1)|, |J(E_2)|, \dots, |J(E_\ell)|, |J(E')|\}$  and  $r = \text{rk } T(A)$ . Then  $c(A^{\otimes k}) \leq \max(r, m)^k \min(r, m)$*

**Proof.** By taking a star-like network topology (as in (10)) we get an execution with  $a(\mathcal{T}_D) = \max(r, m)$  and cost  $c(\mathcal{T}_D) = m \cdot r$ . ◀

## 5 A lower bound for the cost of a multilinear map

In this section, we prove a general lower bound on the cost of evaluating a multilinear map using tensor networks, as defined in Section 3. The lower bound is expressed in terms of the *socket-width* of a multilinear map, which we now proceed to define.

Let  $A : \mathbb{F}^{J(E_1)} \times \mathbb{F}^{J(E_2)} \times \dots \times \mathbb{F}^{J(E_\ell)} \rightarrow \mathbb{F}^{J(E')}$  be an  $\ell$ -linear map. A *socket-tree* of  $A$  is a tree  $\mathcal{T}_S$  whose  $\ell + 1$  leaf vertices are the sockets  $E_1, E_2, \dots, E_\ell, E'$  of  $A$  and whose internal vertices all have degree exactly 3. Associate with each edge  $e = \{x_R, x_C\}$  of  $\mathcal{T}_S$  the two subtrees  $\mathcal{T}_S(x_R, e)$  and  $\mathcal{T}_S(x_C, e)$  obtained by removing  $e$ , where  $\mathcal{T}_S(x_R, e)$  is the subtree containing  $x_R$  and  $\mathcal{T}_S(x_C, e)$  is the subtree containing  $x_C$ . Let  $L(x_R, e)$  be the set of leaves in  $\mathcal{T}_S(x_R, e)$  and let  $L(x_C, e)$  be the set of leaves in  $\mathcal{T}_S(x_C, e)$ .

The sets  $L(x_R, e)$  and  $L(x_C, e)$  are both nonempty and together partition the set of sockets. Consider the flattening  $M(\mathcal{T}_S, e)$  of the tensor  $T(A)$  such that the modes in  $L(x_R, e)$  index the rows and the modes in  $L(x_C, e)$  index the columns of  $M(\mathcal{T}_S, e)$ . The *width* of  $\mathcal{T}_S$  at  $e$  is the rank of  $M(\mathcal{T}_S, e)$ , and the *width* of  $\mathcal{T}_S$  is  $w(\mathcal{T}_S) = \max_{e \in E(\mathcal{T}_S)} \text{rk}(M(\mathcal{T}_S, e))$ .

Let us write  $\mathcal{S}(A)$  for the set of all socket-trees of the multilinear form  $A$ . We define the *socket-width* of  $A$  to be  $w(A) = \min_{\mathcal{T}_S \in \mathcal{S}(A)} w(\mathcal{T}_S)$ .

The rest of this section is devoted to proving Theorem 1.4:

► **Theorem 1.4.** *For every multilinear map  $A$ , it holds that  $c(A) \geq w(A)$ .*

First, we prove that without loss of generality, we may restrict our attention to forms.

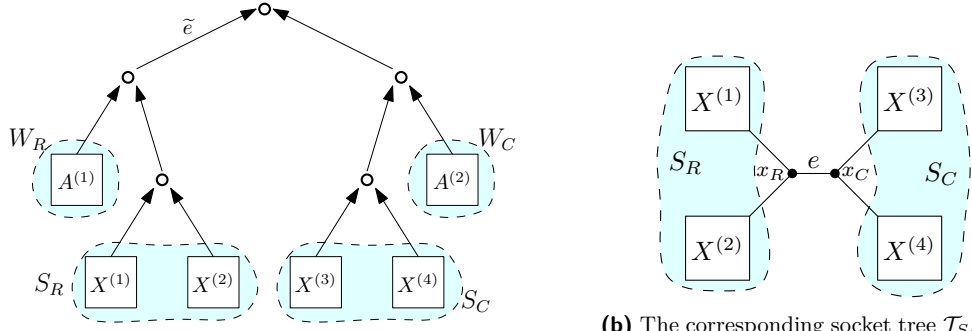
► **Claim 5.1.** *For any multilinear map  $A$ , it holds that  $c(A) \geq c(F(A))$ .*

**Proof.** We observe that  $A$  and  $F(A)$  satisfy  $\hat{T}(A) = \hat{T}(F(A))$ . Any network  $D \in \mathcal{D}(A)$  can be modified to a network  $D' \in \mathcal{D}(F(A))$  by attaching a tensor  $X' \in \mathbb{F}^{J(E')}$  to the boundary of  $D$ . Let  $D \in \mathcal{D}(A)$  be such that  $c(D) = c(A)$ . The minimum-cost execution of  $D$ , followed by contracting  $T(D)$  and  $X'$ , is an execution of  $D'$ . Its cost is  $c(A)$ , since the cost of contracting  $T(D)$  and  $X'$  is  $\prod_{e \in B(D)} |J(e)|$  and  $\prod_{e \in B(D)} |J(e)| \leq c(A)$ , because the last step of the minimum-cost execution of  $D$  contracted a set  $W$  with all modes  $e \in B(D)$  incident to  $W$ . Thus,  $c(A) \geq c(F(A))$ . ◀

Furthermore,  $w(A) = w(F(A))$  for every multilinear map  $A$ , since  $w(A)$  only depends on the tensor  $T(A)$ , but not on which of its coordinates (if any) is the output. Thus it suffices to prove Theorem 1.4 for multilinear forms, which we now proceed to do.

► **Lemma 5.2.** *For any multilinear form  $F$ , it holds that  $c(F) \geq w(F)$ .*

**Proof.** Let  $D \in \mathcal{D}(F)$  be such that  $c(D) = c(F)$ . It is a tensor network with empty boundary and a socket vertex  $S_i \in V(D)$  for each input socket  $E_i$ , where  $i = 1, 2, \dots, \ell$ . Its tensor is  $T(D) = F(X^{(1)}, X^{(2)}, \dots, X^{(\ell)})$  where  $X^{(i)} = T(S_i)$  for  $i = 1, 2, \dots, \ell$ .



(a) Example of a possible execution tree  $\mathcal{T}_D$ . Given the choice of  $e$  in the corresponding socket tree  $\mathcal{T}_S$  shown on the right there are four possible choices of  $\tilde{e}$ .

(b) The corresponding socket tree  $\mathcal{T}_S$ . The exact choice of  $\tilde{e}$  in  $\mathcal{T}_D$  determines which part of the cut is the  $x_R$  part, and which is the  $x_C$  part.

■ **Figure 1** Illustration of the notation used for the execution and socket trees.

By Lemma 3.1, a minimum-cost execution of  $D$  can be represented by a rooted binary tree  $\mathcal{T}_D$ , where the set of leaves of  $\mathcal{T}_D$  are  $V(D)$  and each inner vertex represents the vertex obtained by contracting its two children. Let  $\mathcal{T}_S$  be the unique socket-tree of  $F$  that is obtained as a topological minor of  $\mathcal{T}_D$ . Slightly abusing the notation, we assume that the leaves of  $\mathcal{T}_S$  are the socket vertices  $S_1, S_2, \dots, S_\ell$  instead of the sockets  $E_1, E_2, \dots, E_\ell$ . To establish the lemma, it suffices to show that  $\mathcal{T}_D$  has cost at least  $w(\mathcal{T}_S)$ , since  $w(\mathcal{T}_S) \geq w(F)$ .

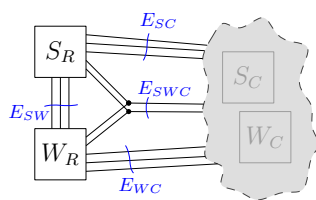
Let  $e = \{x_R, x_C\} \in E(\mathcal{T}_S)$  be an edge of the socket tree  $\mathcal{T}_S$  with  $\text{rk}(M(\mathcal{T}_S, e)) = w(\mathcal{T}_S)$ , and let  $\tilde{e}$  be an edge of the execution tree  $\mathcal{T}_D$  in the subdivision of  $e$  appearing in  $\mathcal{T}_D$ . Without loss of generality we may assume that  $\tilde{e}$  is directed from the part of  $\mathcal{T}_D$  corresponding to  $x_R$  towards the part corresponding to  $x_C$  (if not, simply switch names of  $x_R$  and  $x_C$ ). Define  $S_R = L(x_R, e)$  and  $S_C = L(x_C, e)$ . Let  $W_R \subseteq V(D)$  be the set of non-socket vertices of  $D$  that appear on the same side of  $\tilde{e}$  in  $\mathcal{T}_D$  with socket vertices  $S_R$  and let  $W_C$  be the set of remaining non-socket vertices of  $D$ . See Figure 1 for an illustration of all these definitions. Finally, let  $D' = D/S_R/S_C/W_R/W_C$  be the result of contracting each of these four sets of vertices of  $D$ . For notational convenience, we identify the four vertices of the new network with the four subsets  $S_R, S_C, W_R, W_C$ .

Now, the tensor  $P = T(D'[W_R \cup S_R])$  appears as an intermediate result in the execution  $\mathcal{T}_D$ ,<sup>5</sup> hence the volume of  $P$  is a lower bound on the cost of  $\mathcal{T}_D$ .

We group the modes of  $D'$  incident on  $S_R$  or  $W_R$  as shown in Figure 2:  $E_{SW}$  are all modes in  $D'$  incident exactly upon  $S_R$  and  $W_R$ ,  $E_{WC}$  are all modes incident on  $W_R$  but not on  $S_R$ ,  $E_{SC}$  are all modes incident on  $S_R$  but not  $W_R$ , and finally  $E_{SWC}$  are all modes incident upon  $S_R, W_R$ , and at least one of  $S_C$  or  $W_C$ . Write  $E_S = E_{SW} \cup E_{SC} \cup E_{SWC}$  for the modes incident on  $S_R$ , and similarly  $E_C = E_{WC} \cup E_{SC} \cup E_{SWC}$  for all modes incident upon at least one of  $S_R/W_R$  and at least one of  $S_C/W_C$ . Note that  $|J(E_C)|$  is precisely the volume of  $P$  which we aim to lower bound.

Define a matrix  $A \in \mathbb{F}^{J(E_S)} \times \mathbb{F}^{J(E_C)}$  as follows. We identify its row indices  $i \in J(E_S)$  as being triples  $i = (i_{SW}, i_{SC}, i_{SWC}) \in J(E_{SW}) \times J(E_{SC}) \times J(E_{SWC})$  and similarly its column indices  $j \in J(E_C)$  are triples  $j = (j_{SC}, j_{WC}, j_{SWC}) \in J(E_{SC}) \times J(E_{WC}) \times J(E_{SWC})$ . Then

<sup>5</sup> Note that the same is not true for the tensor  $T(D'[W_C \cup S_C])$ .



■ **Figure 2** Illustration of  $D'$ . We group the modes of  $D'$  based on how they connect  $S_R$ ,  $S_C$ , and the “ $C$  part” of  $D'$ .

the entries of  $A$  are

$$A_{(i_{SW}, i_{SC}, i_{SWC}), (j_{SC}, j_{WC}, j_{SWC})} = \begin{cases} T(D'[W_R])_{i_{SW}, j_{WC}, j_{SWC}} & \text{if } i_{SC} = j_{SC} \wedge i_{SWC} = j_{SWC}, \\ 0 & \text{otherwise,} \end{cases}$$

In the case when  $E_S = E_{SW}$  (i.e., all modes incident on  $S_R$  connect only to  $W_R$ ),  $A$  is simply a flattening of  $T(D'[W_R])$ . Recall that  $T(D'[S_R]) \in \prod_{e \in E_S} \mathbb{F}^{J(e)}$ . Then for every  $j = (j_{SC}, j_{WC}, j_{SWC}) \in J(E_C)$ , we have

$$\begin{aligned} \sum_{i \in J(E_S)} A_{i,j} T(D'[S_R])_i &= \sum_{i_{SW} \in J(E_{SW})} A_{(i_{SW}, j_{SC}, j_{SWC}), j} T(D'[S_R])_{i_{SW}, j_{SC}, j_{SWC}} \\ &= \sum_{i_{SW}} T(D'[W_R])_{i_{SW}, j_{WC}, j_{SWC}} T(D'[S_R])_{i_{SW}, j_{SC}, j_{SWC}} \\ &= P_{j_{SC}, j_{WC}, j_{SWC}} = P_j \end{aligned}$$

(recall that  $P$  is the contraction of  $T(D'[W_R])$  and  $T(D'[S_R])$ ). Viewing  $T(D'[S_R])$  as a row vector in  $\mathbb{F}^{J(E_S)}$  we see that  $P$  is the vector-matrix product  $P = T(D'[S_R]) \cdot A \in \mathbb{F}^{J(E_C)}$ .

Symmetrically, for the other half of  $D'$ , we can write  $Q = T(D'[W_C \cup S_C])$  as a matrix-vector product  $Q = B \cdot T(D'[S_C]) \in \mathbb{F}^{J(E_C)}$  where  $B$  is a matrix corresponding to  $T(D'[W_S])$  analogously to how  $A$  corresponds to  $T(D'[W_R])$ .

Thus we have  $T(D) = T(D'[S_R]) \cdot A \cdot B \cdot T(D'[S_C])$ . Recall that for each socket vertex  $S_i$  in the original network  $D$ , we have  $T(S_i) = X^{(i)}$ . Denoting  $X_R = T(D'[S_R])$  and  $X_C = T(D'[S_C])$ , we get  $X_R = \bigotimes_{S_i \in S_R} X^{(i)}$  and  $X_C = \bigotimes_{S_i \in S_C} X^{(i)}$ .<sup>6</sup> Hence

$$F(X^{(1)}, X^{(2)}, \dots, X^{(\ell)}) = X_R \cdot A \cdot B \cdot X_C.$$

It follows that  $A \cdot B$  is the flattening of  $T(F)$  to a matrix with rows indexed by the sockets in  $S_R$  and columns indexed by the sockets in  $S_C$ . But this flattening is precisely the matrix  $M(\mathcal{T}_S, e)$ , implying that  $|J(E_C)| \geq \text{rk}(M(\mathcal{T}_S, e)) = w(\mathcal{T}_S)$ , as desired. ◀

## References

- 1 Amir Abboud, Arturs Backurs, Karl Bringmann, and Marvin Künnemann. Fine-Grained Complexity of Analyzing Compressed Data: Quantifying Improvements over Decompress-and-Solve. In *Proc. of Foundations of Computer Science (FOCS)*, pages 192–203, 2017. doi:10.1109/FOCS.2017.26.

<sup>6</sup> These identities use the fact that  $D$  is derived from a *non-degenerate* network  $D^*$  for  $\hat{T}(F)$ . In particular, every mode in the network  $D$  is incident upon at least one non-socket vertex, hence all modes incident upon  $S_R$  are boundary modes in  $D'[S_R]$ , and similarly for  $S_C$ .

- 2 Amir Abboud, Arturs Backurs, and Virginia Vassilevska Williams. If the Current Clique Algorithms are Optimal, So is Valiant's Parser. In *Proc. of Foundations of Computer Science (FOCS)*, pages 98–117, 2015. doi:10.1109/FOCS.2015.16.
- 3 Michael Alekhovich, Allan Borodin, Joshua Buresh-Oppenheimer, Russell Impagliazzo, Avner Magen, and Toniann Pitassi. Toward a Model for Backtracking and Dynamic Programming. *Comput. Complex.*, 20(4):679–740, 2011. doi:10.1007/s00037-011-0028-y.
- 4 Noga Alon and Shai Gutner. Balanced families of perfect hash functions and their applications. *ACM Trans. Algorithms*, 6(3):54:1–54:12, 2010. doi:10.1145/1798596.1798607.
- 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 Itai Arad and Zeph Landau. Quantum computation and the evaluation of tensor networks. *SIAM J. Comput.*, 39(7):3089–3121, 2010. doi:10.1137/080739379.
- 7 Sanjeev Arora and Boaz Barak. *Computational Complexity: A Modern Approach*. Cambridge University Press, New York, NY, USA, 1st edition, 2009. doi:10.1017/CB09780511804090.
- 8 Martin Beaudry and Markus Holzer. The Complexity of Tensor Circuit Evaluation. *Comput. Complex.*, 16(1):60–111, 2007. doi:10.1007/s00037-007-0222-0.
- 9 Stuart J. Berkowitz. On Computing the Determinant in Small Parallel Time Using a Small Number of Processors. *Inf. Process. Lett.*, 18(3):147–150, 1984. doi:10.1016/0020-0190(84)90018-8.
- 10 Andreas Björklund, Thore Husfeldt, Petteri Kaski, and Mikko Koivisto. Fourier meets Möbius: fast subset convolution. In *Proc. of Symposium on Theory of Computing (STOC)*, pages 67–74, 2007. doi:10.1145/1250790.1250801.
- 11 Andreas Björklund, Thore Husfeldt, Petteri Kaski, and Mikko Koivisto. Computing the Tutte Polynomial in Vertex-Exponential Time. In *Proc. of Foundations of Computer Science (FOCS)*, pages 677–686, 2008. doi:10.1109/FOCS.2008.40.
- 12 Andreas Björklund, Thore Husfeldt, Petteri Kaski, and Mikko Koivisto. Counting Paths and Packings in Halves. In *Proc. of European Symposium on Algorithms (ESA)*, pages 578–586, 2009. doi:10.1007/978-3-642-04128-0\_52.
- 13 Andreas Björklund, Thore Husfeldt, and Mikko Koivisto. Set Partitioning via Inclusion-Exclusion. *SIAM J. Comput.*, 39(2):546–563, 2009. doi:10.1137/070683933.
- 14 Andreas Björklund, Petteri Kaski, and Łukasz Kowalik. Counting Thin Subgraphs via Packings Faster Than Meet-in-the-Middle Time. In *Proc. of Symposium on Discrete Algorithms (SODA)*, pages 594–603, 2014. doi:10.1137/1.9781611973402.45.
- 15 Hans L. Bodlaender, Erik Jan van Leeuwen, Johan M. M. van Rooij, and Martin Vatshelle. Faster Algorithms on Branch and Clique Decompositions. In *Proc. of Mathematical Foundations of Computer Science (MFCS)*, pages 174–185, 2010. doi:10.1007/978-3-642-15155-2\_17.
- 16 James R. Bunch and John E. Hopcroft. Triangular factorization and inversion by fast matrix multiplication. *Math. Comp.*, 28:231–236, 1974. doi:10.2307/2005828.
- 17 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.
- 18 Jin-yi Cai, Pinyan Lu, and Mingji Xia. Computational Complexity of Holant Problems. *SIAM J. Comput.*, 40(4):1101–1132, 2011. doi:10.1137/100814585.
- 19 Florent Capelli, Arnaud Durand, and Stefan Mengel. The Arithmetic Complexity of Tensor Contraction. *Theory Comput. Syst.*, 58(4):506–527, 2016. doi:10.1007/s00224-015-9630-8.
- 20 A. Cayley. On the theory of the analytical forms called trees. *Philos. Mag.*, 13:172–176, 1857. doi:10.1080/14786445708642275.

- 21 A. Cayley. On the analytical forms called trees, part II. *Philos. Mag.*, 18:374–378, 1859. doi:10.1080/14786445908642782.
- 22 A. Clebsch. Ueber symbolische Darstellung algebraischer Formen. *J. Reine Angew. Math.*, 59:1–62, 1861. doi:10.1515/crll.1861.59.1.
- 23 W. Clifford. Extract of a Letter to Mr. Sylvester from Prof. Clifford of University College, London. *Amer. J. Math.*, 1(2):126–128, 1878. doi:10.2307/2369303.
- 24 James W. Cooley and John W. Tukey. An algorithm for the machine calculation of complex Fourier series. *Math. Comp.*, 19:297–301, 1965. doi:10.2307/2003354.
- 25 Radu Curticapean, Holger Dell, and Dániel Marx. Homomorphisms are a good basis for counting small subgraphs. In *Proc. of Symposium on Theory of Computing (STOC)*, pages 210–223, 2017. doi:10.1145/3055399.3055502.
- 26 Carsten Damm, Markus Holzer, and Pierre McKenzie. The complexity of tensor calculus. *Comput. Complex.*, 11(1-2):54–89, 2002. doi:10.1007/s00037-000-0170-4.
- 27 Sashka Davis and Russell Impagliazzo. Models of Greedy Algorithms for Graph Problems. *Algorithmica*, 54(3):269–317, 2009. doi:10.1007/s00453-007-9124-4.
- 28 Josep Díaz, Maria J. Serna, and Dimitrios M. Thilikos. Counting H-colorings of partial k-trees. *Theor. Comput. Sci.*, 281(1-2):291–309, 2002. doi:10.1016/S0304-3975(02)00017-8.
- 29 Frederic Dorn. Dynamic Programming and Fast Matrix Multiplication. In *Proc. of European Symposium on Algorithms (ESA)*, pages 280–291, 2006. doi:10.1007/11841036\_27.
- 30 Friedrich Eisenbrand and Fabrizio Grandoni. On the complexity of fixed parameter clique and dominating set. *Theoret. Comput. Sci.*, 326(1-3):57–67, 2004. doi:10.1016/j.tcs.2004.05.009.
- 31 Norman P. Jouppi et al. In-Datacenter Performance Analysis of a Tensor Processing Unit. In *Proc. of International Symposium on Computer Architecture (ISCA)*, pages 1–12, 2017. doi:10.1145/3079856.3080246.
- 32 Peter Floderus, Miroslaw Kowaluk, Andrzej Lingas, and Eva-Marta Lundell. Detecting and Counting Small Pattern Graphs. *SIAM J. Discrete Math.*, 29(3):1322–1339, 2015. doi:10.1137/140978211.
- 33 Fedor V. Fomin, Daniel Lokshantov, Venkatesh Raman, Saket Saurabh, and B. V. Raghavendra Rao. Faster algorithms for finding and counting subgraphs. *J. Comput. Syst. Sci.*, 78(3):698–706, 2012. doi:10.1016/j.jcss.2011.10.001.
- 34 William I. Gasarch. The Second P =? NP Poll. *SIGACT News Complexity Theory Column*, 74, 2012. doi:10.1145/2261417.2261434.
- 35 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.
- 36 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.
- 37 Alon Itai and Michael Rodeh. Finding a Minimum Circuit in a Graph. *SIAM J. Comput.*, 7(4):413–423, 1978. doi:10.1137/0207033.
- 38 Matti Karppa, Petteri Kaski, and Jukka Kohonen. A Faster Subquadratic Algorithm for Finding Outlier Correlations. *ACM Trans. Algorithms*, 14(3):31:1–31:26, 2018. doi:10.1145/3174804.
- 39 L. H. Kauffman. Knots, abstract tensors and the Yang-Baxter equation. In *Knots, topology and quantum field theories*, pages 179–334. World Scientific, 1989.
- 40 A. B. Kempe. On the Application of Clifford’s Graphs to Ordinary Binary Quantics. *P. Lond. Math. Soc.*, 17:107–121, 1885/86. doi:10.1112/plms/s1-17.1.107.

- 41 A. B. Kempe. On the application of the Sylvester-Clifford Graphs to Ordinary Binary Quantics. (Second Part.). *P. Lond. Math. Soc.*, 24:97–118, 1892/93. doi:10.1112/plms/s1-24.1.97.
- 42 Ton Kloks, Dieter Kratsch, and Haiko Müller. Finding and counting small induced subgraphs efficiently. *Inf. Process. Lett.*, 74(3-4):115–121, 2000. doi:10.1016/S0020-0190(00)00047-8.
- 43 Tamara G. Kolda. Multilinear operators for higher-order decompositions. Technical Report SAND2006-2081, Sandia National Laboratories, 2006. doi:10.2172/923081.
- 44 Tamara G. Kolda and Brett W. Bader. Tensor decompositions and applications. *SIAM Rev.*, 51(3):455–500, 2009. doi:10.1137/07070111X.
- 45 Daphne Koller and Nir Friedman. *Probabilistic Graphical Models*. Adaptive Computation and Machine Learning. MIT Press, Cambridge, MA, 2009.
- 46 Joseph B. Kruskal. Three-way arrays: rank and uniqueness of trilinear decompositions, with application to arithmetic complexity and statistics. *Linear Algebra Appl.*, 18(2):95–138, 1977. doi:10.1016/0024-3795(77)90069-6.
- 47 Greg Kuperberg. Involutory Hopf algebras and 3-manifold invariants. *Internat. J. Math.*, 2(1):41–66, 1991. doi:10.1142/S0129167X91000053.
- 48 J. M. Landsberg. *Tensors: Geometry and Applications*, volume 128 of *Graduate Studies in Mathematics*. American Mathematical Society, Providence, RI, 2012. doi:10.1090/gsm/128.
- 49 Joseph M. Landsberg, Yang Qi, and Ke Ye. On the geometry of tensor network states. *Quantum Inf. Comput.*, 12(3-4):346–354, 2012.
- 50 François Le Gall. Powers of tensors and fast matrix multiplication. In *Proc. of International Symposium on Symbolic and Algebraic Computation (ISSAC)*, pages 296–303, 2014. doi:10.1145/2608628.2608664.
- 51 James R. Lee, Prasad Raghavendra, and David Steurer. Lower Bounds on the Size of Semidefinite Programming Relaxations. In *Proc. of Symposium on Theory of Computing (STOC)*, pages 567–576, 2015. doi:10.1145/2746539.2746599.
- 52 Andrea Lincoln, Virginia Vassilevska Williams, and R. Ryan Williams. Tight Hardness for Shortest Cycles and Paths in Sparse Graphs. In *Proc. of Symposium on Discrete Algorithms (SODA)*, pages 1236–1252, 2018. doi:10.1137/1.9781611975031.80.
- 53 Stefano Markidis, Steven Wei Der Chien, Erwin Laure, Ivy Bo Peng, and Jeffrey S. Vetter. NVIDIA tensor core programmability, performance & precision. In *Proc. of International Parallel and Distributed Processing Symposium Workshops (IPDPS)*, pages 522–531, 2018. doi:10.1109/IPDPSW.2018.00091.
- 54 J. Nešetřil and S. Poljak. On the complexity of the subgraph problem. *Comment. Math. Univ. Carolin.*, 26(2):415–419, 1985.
- 55 Stephan Olariu. Paw-Free Graphs. *Inf. Process. Lett.*, 28(1):53–54, 1988. doi:10.1016/0020-0190(88)90143-3.
- 56 R. Orús. A practical introduction to tensor networks: Matrix product states and projected entangled pair states. *Ann. Phys.*, 349:117–158, 2014. doi:10.1016/j.aop.2014.06.013.
- 57 Roger Penrose. *Tensor Methods in Algebraic Geometry*. PhD thesis, St. John’s College, 1956.
- 58 Roger Penrose. Applications of negative dimensional tensors. In *Combinatorial Mathematics and its Applications*, pages 221–244. Academic Press, 1971.
- 59 Robert N. C. Pfeifer, Jutho Haegeman, and Frank Verstraete. Faster identification of optimal contraction sequences for tensor networks. *Phys. Rev. E*, 90:033315, 2014. doi:10.1103/PhysRevE.90.033315.
- 60 Ran Raz. Tensor-Rank and Lower Bounds for Arithmetic Formulas. *J. ACM*, 60(6):40:1–40:15, 2013. doi:10.1145/2535928.



- 61 Jürgen Richter-Gebert and Peter Lebmeir. Diagrams, tensors and geometric reasoning. *Discrete Comput. Geom.*, 42(2):305–334, 2009. doi:10.1007/s00454-009-9188-9.
- 62 Elina Robeva and Anna Seigal. Duality of graphical models and tensor networks. *Inf. Inference*, page iay009, 2018. doi:10.1093/imaiai/iay009.
- 63 Günter Rote. Division-Free Algorithms for the Determinant and the Pfaffian: Algebraic and Combinatorial Approaches. In *Computational Discrete Mathematics, Advanced Lectures*, pages 119–135, 2001. doi:10.1007/3-540-45506-X\_9.
- 64 Herbert John Ryser. *Combinatorial Mathematics*. The Carus Mathematical Monographs, No. 14. Mathematical Association of America, 1963.
- 65 Alexander Schrijver. On traces of tensor representations of diagrams. *Linear Algebra Appl.*, 476:28–41, 2015. doi:10.1016/j.laa.2015.02.037.
- 66 Edgar Solomonik and Torsten Hoefer. Sparse Tensor Algebra as a Parallel Programming Model. *arXiv:1512.00066*, 2015.
- 67 Edgar Solomonik, Devin Matthews, Jeff R. Hammond, John F. Stanton, and James Demmel. A massively parallel tensor contraction framework for coupled-cluster computations. *J. Parallel Distrib. Comput.*, 74(12):3176–3190, 2014. doi:10.1016/j.jpdc.2014.06.002.
- 68 Volker Strassen. Gaussian elimination is not optimal. *Numer. Math.*, 13:354–356, 1969. doi:10.1007/BF02165411.
- 69 J. J. Sylvester. On an Application of the New Atomic Theory to the Graphical Representation of the Invariants and Covariants of Binary Quantics, with Three Appendices. *Amer. J. Math.*, 1(1):64–125, 1878. doi:10.2307/2369436.
- 70 Leslie G. Valiant. The complexity of computing the permanent. *Theor. Comput. Sci.*, 8:189–201, 1979. doi:10.1016/0304-3975(79)90044-6.
- 71 Leslie G. Valiant. Holographic Algorithms. *SIAM J. Comput.*, 37(5):1565–1594, 2008. doi:10.1137/070682575.
- 72 Charles Van Loan. *Computational Frameworks for the Fast Fourier Transform*. SIAM, 1992. doi:10.1137/1.9781611970999.
- 73 Virginia Vassilevska Williams. Multiplying matrices faster than Coppersmith-Winograd. In *Proc. of Symposium on Theory of Computing (STOC)*, pages 887–898, 2012. doi:10.1145/2213977.2214056.
- 74 Virginia Vassilevska Williams and Ryan Williams. Finding, Minimizing, and Counting Weighted Subgraphs. *SIAM J. Comput.*, 42(3):831–854, 2013. doi:10.1137/09076619X.
- 75 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.
- 76 Ryan Williams. Faster all-pairs shortest paths via circuit complexity. In *Proc. of Symposium on Theory of Computing (STOC)*, pages 664–673, 2014. doi:10.1145/2591796.2591811.
- 77 Virginia Vassilevska Williams, Joshua R. Wang, Richard Ryan Williams, and Huacheng Yu. Finding Four-Node Subgraphs in Triangle Time. In *Proc. of Symposium on Discrete Algorithms (SODA)*, pages 1671–1680, 2015. doi:10.1137/1.9781611973730.111.





# A #SAT Algorithm for Small Constant-Depth Circuits with PTF Gates

**Swapnam Bajpai**

Indian Institute of Technology, Bombay, Mumbai, India  
swapnam@cse.iitb.ac.in

**Vaibhav Krishan**

Indian Institute of Technology, Bombay, Mumbai, India  
vaibhkrishan@iitb.ac.in

**Deepanshu Kush**

Indian Institute of Technology, Bombay, Mumbai, India  
kush.deepanshu@gmail.com

**Nutan Limaye<sup>1</sup>**

Indian Institute of Technology, Bombay, Mumbai, India  
nutan@cse.iitb.ac.in

**Srikanth Srinivasan<sup>2</sup>**

Department of Mathematics, Indian Institute of Technology Bombay, Mumbai, India  
srikanth@math.iitb.ac.in

---

## Abstract

We show that there is a zero-error randomized algorithm that, when given a small constant-depth Boolean circuit  $C$  made up of gates that compute constant-degree Polynomial Threshold functions or PTFs (i.e., Boolean functions that compute signs of constant-degree polynomials), counts the number of satisfying assignments to  $C$  in significantly better than brute-force time.

Formally, for any constants  $d, k$ , there is an  $\varepsilon > 0$  such that the zero-error randomized algorithm counts the number of satisfying assignments to a given depth- $d$  circuit  $C$  made up of  $k$ -PTF gates such that  $C$  has size at most  $n^{1+\varepsilon}$ . The algorithm runs in time  $2^{n-n^{\Omega(\varepsilon)}}$ .

Before our result, no algorithm for beating brute-force search was known for counting the number of satisfying assignments even for a single degree- $k$  PTF (which is a depth-1 circuit of linear size).

The main new tool is the use of a learning algorithm for learning degree-1 PTFs (or Linear Threshold Functions) using comparison queries due to Kane, Lovett, Moran and Zhang (FOCS 2017). We show that their ideas fit nicely into a memoization approach that yields the #SAT algorithms.

**2012 ACM Subject Classification** Theory of computation → Circuit complexity

**Keywords and phrases** SAT, Polynomial Threshold Functions, Constant-depth Boolean Circuits, Linear Decision Trees, Zero-error randomized algorithms

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2019.8

**Acknowledgements** We would like to thank Valentine Kabanets for telling us about this problem and Paul Beame, Daniel Kane, Valentine Kabanets, Ryan O'Donnell, Rahul Santhanam, Madhu Sudan and Ryan Williams for helpful discussions. We would also like to thank the anonymous referees of ITCS 2019 for their helpful comments and suggestions.

---

<sup>1</sup> Funded by Mathematical Research Impact Centric Support (MATRICS), SERB.

<sup>2</sup> Funded by Mathematical Research Impact Centric Support (MATRICS), SERB. Work partially done while visiting the Simons Institute for the Theory of Computing, Berkeley.



## 1 Introduction

This paper adds to the growing line of work on *circuit-analysis algorithms*, where we are given as input a Boolean circuit  $C$  from a fixed class  $\mathcal{C}$  computing a function  $f : \{-1, 1\}^n \rightarrow \{-1, 1\}$ ,<sup>3</sup> and we are required to compute some parameter of the function  $f$ . A typical example of this is the question of satisfiability, i.e. whether  $f$  is the constant function 1 or not. In this paper, we are interested in computing  $\#SAT(f)$ , which is the number of satisfying assignments of  $f$  (i.e.  $|\{a \in \{-1, 1\}^n \mid f(a) = -1\}|$ ).

Problems of this form can always be solved by “brute-force” in time  $\text{poly}(|C|) \cdot 2^n$  by trying all assignments to  $C$ . The question is can this brute-force algorithm be significantly improved, say to time  $2^n/n^{\omega(1)}$  when  $C$  is small, say  $|C| \leq n^{O(1)}$ .

Such algorithms, intuitively are able to distinguish a small circuit  $C \in \mathcal{C}$  from a “black-box” and hence find some structure in  $C$ . This structure, in turn, is useful in answering other questions about  $\mathcal{C}$ , such as proving lower bounds against the class  $\mathcal{C}$ .<sup>4</sup> There has been a large body of work in this area, a small sample of which can be found in [21, 20, 26, 27]. A striking result of this type was proved by Williams [26] who showed that for many circuit classes  $\mathcal{C}$ , even *co-non-deterministic* satisfiability algorithms running in better than brute-force time yield lower bounds against  $\mathcal{C}$ .

Recently, researchers have also uncovered tight connections between many combinatorial problems and circuit-analysis algorithms, showing that even modest improvements over brute-force search can be used to improve long-standing bounds for these combinatorial problems (see, e.g., [30, 2, 3, 1]). This yields further impetus in improving known circuit-analysis algorithms.

This paper is concerned with #SAT algorithms for constant depth threshold circuits, denoted as  $TC^0$ , which are Boolean circuits where each gate computes a linear threshold function (LTF); an LTF computes a Boolean function which accepts or rejects based on the sign of a (real-valued) linear polynomial evaluated on its input. Such circuits are surprisingly powerful: for example, they can perform all integer arithmetic efficiently [4, 9], and are at the frontier of our current lower bound techniques [16, 5].

It is natural, therefore, to try to come up with circuit-analysis algorithms for threshold circuits. Indeed, there has a large body of work in the area (reviewed below), but some extremely simple questions remain open.

An example of such a question is the existence of a better-than-brute-force algorithm for satisfiability of degree- $k$  PTFs where  $k$  is a constant greater than 1. Informally, the question is the following: we are given a degree- $k$  polynomial  $Q(x_1, \dots, x_n)$  in  $n$  Boolean variables and we ask if there is any Boolean assignment  $a \in \{-1, 1\}^n$  to  $x_1, \dots, x_n$  such that  $Q(a) < 0$ . (Note that for a linear polynomial (i.e.  $k = 1$ ), this problem is trivial.)

Surprisingly, no algorithm is known for this problem that is significantly better than  $2^n$  time.<sup>5</sup> In this paper, we solve the stronger counting variant of this problem for any constant-degree PTFs. We start with some definitions and then describe this result.

► **Definition 1** (Polynomial Threshold Functions). A *Polynomial Threshold Function (PTF)* on  $n$  variables of degree- $k$  is a Boolean function  $f : \{-1, 1\}^n \rightarrow \{-1, 1\}$  such that there is

<sup>3</sup> We work with the  $\{-1, 1\}$  basis for Boolean functions, which is by now standard in the literature. (See for instance [18].) Here  $-1$  stands for True and  $1$  stands for False.

<sup>4</sup> This intuition was provided to us by Ryan Williams.

<sup>5</sup> An algorithm was claimed for this problem in the work of Sakai, Seto, Tamaki and Teruyama [22]. Unfortunately, the proof of this claim only works when the weights are suitably small. See Footnote 1 on page 4 of [14].

a degree- $k$  multilinear polynomial  $P(x_1, \dots, x_n) \in \mathbb{R}[x_1, \dots, x_n]$  that, for all  $a \in \{-1, 1\}^n$ , satisfies  $f(a) = \text{sgn}(P(a))$ . (We assume that  $P(a) \neq 0$  for any  $a \in \{-1, 1\}^n$ .)

In such a scenario, we call  $f$  a  $k$ -PTF. In the special case that  $k = 1$ , we call  $f$  a *Linear Threshold function (LTF)*. We also say that the polynomial  $P$  *sign-represents*  $f$ .

When  $P \in \mathbb{Z}[x_1, \dots, x_n]$ , we define the weight of  $P$ , denoted  $w(P)$ , to be the *bit-complexity* of the sum of the absolute values of all the coefficients of  $P$ . In particular, the coefficients of  $P$  are integers in the range  $[-2^{w(P)}, 2^{w(P)}]$ .

We now formally define the #SAT problem for  $k$ -PTFs. Throughout, we assume that  $k$  is a constant and not a part of the input.

► **Definition 2** (#SAT problem for  $k$ -PTFs). The problem is defined as follows.

**Input:** A  $k$ -PTF  $f$ , specified by a degree- $k$  polynomial  $P(x_1, \dots, x_n)$  with integer coefficients.<sup>6</sup>

**Output:** The number of satisfying assignments to  $f$ . That is, the number of  $a \in \{-1, 1\}^n$  such that  $P(a) < 0$ .

We use  $\#SAT(f)$  to denote this output. We say that the input instance has parameters  $(n, M)$  if  $n$  is the number of input variables and  $w(P) \leq M$ .

► **Remark.** An interesting setting of  $M$  is  $\text{poly}(n)$  since any  $k$ -PTF can be represented by an integer polynomial with coefficients of bit-complexity at most  $\tilde{O}(n^k)$  [17]. However, note that our algorithms are even when  $M$  is  $\exp(n^{o(1)})$ , i.e. when the weights are slightly short of doubly exponential in  $n$ .

We give a better-than-brute-force algorithm for #SAT( $k$ -PTF). Formally we prove the following theorem.

► **Theorem 3.** *Fix any constant  $k$ . There is a zero-error randomized algorithm that solves the #SAT problem for  $k$ -PTFs in time  $\text{poly}(n, M) \cdot 2^{n-S}$  where  $S = \tilde{\Omega}(n^{1/(k+1)})$  and  $(n, M)$  are the parameters of the input  $k$ -PTF  $f$ . (The  $\tilde{\Omega}(\cdot)$  hides factors that are inverse polylogarithmic in  $n$ .)*

► **Remark.** An anonymous ITCS 2019 referee pointed out to us that from two results of Williams [25, 28], it follows that satisfiability for 2-PTFs can be solved in  $2^{n-\Omega(\sqrt{n})}$  time. Note that this is better than the runtime of our algorithm. However, the method does not extend to  $k \geq 3$ .

We then extend this result to a powerful model of circuits called  *$k$ -PTF circuits*, where each gate computes a  $k$ -PTF. This model was first studied by Kane, Kabanets and Lu [13] who proved strong average case lower bounds for slightly superlinear-size constant-depth  $k$ -PTF circuits. Using these ideas, Kabanets and Lu [14] were able to give a #SAT algorithm for a restricted class of  $k$ -PTF circuits, where each gate computes a PTF with a subquadratically many, say  $n^{1.99}$ , monomials (while the size remains the same, i.e. slightly superlinear).<sup>7</sup> A reason for this restriction on the PTFs was that they did not have an algorithm to handle even a single degree-2 PTF (which can have  $\Omega(n^2)$  many monomials).

Building on our #SAT algorithm for  $k$ -PTFs and the ideas of [14], we are able to handle general  $k$ -PTF circuits of slightly superlinear size. We state these results formally below.

We first define  $k$ -PTF circuits formally.

<sup>6</sup> It is known [17] that such a representation always exists.

<sup>7</sup> Their result also works for the slightly larger class of PTFs that are subquadratically sparse in the  $\{0, 1\}$ -basis with *no restriction* on degree. Our result can also be stated for the larger class of *polynomially sparse* PTFs, but for the sake of simplicity, we stick to constant-degree PTFs.

► **Definition 4** (*k*-PTF circuits). A *k*-PTF circuit on  $n$  variables is a Boolean circuit on  $n$  variables where each gate  $g$  of fan-in  $m$  computes a fixed *k*-PTF of its  $m$  inputs. The size of the circuit is the *number of wires* in the circuit, and the depth of the circuit is the longest path from an input to the output gate.<sup>8</sup>

The problems we consider is the #SAT problem for *k*-PTF circuits, defined as follows.

► **Definition 5** (#SAT problem for *k*-PTF circuits). The problem is defined as follows.

**Input:** A *k*-PTF circuit  $C$ , where each gate  $g$  is labelled by an integer polynomial that sign-represents the function that is computed by  $g$ .

**Output:** The number of satisfying assignments to  $C$ .

We use #SAT( $C$ ) to denote this output. We say that the input instance has parameters  $(n, s, d, M)$  where  $n$  is the number of input variables,  $s$  is the size of  $C$ ,  $d$  is the depth of  $C$  and  $M$  is the maximum over the weights of the degree- $k$  polynomials specifying the *k*-PTFs in  $C$ . We will say that  $M$  is the weight of  $C$ , denoted by  $w(C)$ .

We now state our result on #SAT for *k*-PTF circuits. The following result implies Theorem 3, but we prove them separately.

► **Theorem 6.** *Fix any constants  $k, d$ . Then the following holds for some constant  $\varepsilon_{k,d} > 0$  depending on  $k, d$ . There is a zero-error randomized algorithm that solves the #SAT problem for *k*-PTF circuits of size at most  $s = n^{1+\varepsilon_{k,d}}$  with probability at least  $1/4$  and outputs ? otherwise. The algorithm runs in time  $\text{poly}(n, M) \cdot 2^{n-S}$ , where  $S = n^{\varepsilon_{k,d}}$  and  $(n, s, d, M)$  are the parameters of the input *k*-PTF circuit.*

**Previous work.** Satisfiability algorithms for  $\text{TC}^0$  have been widely investigated. Impagliazzo, Lovett, Paturi and Schneider [12, 10] give algorithms for checking satisfiability of depth-2 threshold circuits with  $O(n)$  gates. An incomparable result was proved by Williams [29] who obtained algorithms for subexponential-sized circuits from the class  $\text{ACC}^0 \circ \text{LTF}$ , which is a subclass of subexponential  $\text{TC}^0$ .<sup>9</sup> For the special case of *k*-PTFs (and generalizations to sparse PTFs over the  $\{0, 1\}$  basis) with *small weights*, a #SAT algorithm was devised by Sakai et al. [22].<sup>10</sup> The high-level idea of our algorithm is the same as theirs.

For general constant-depth threshold circuits, the first satisfiability algorithm was given by Chen, Santhanam and Srinivasan [7]. In their paper, Chen et al. gave the first average case lower bound for  $\text{TC}^0$  circuits of slightly super linear size  $n^{1+\varepsilon_d}$ , where  $\varepsilon_d$  depends on the depth of the circuit. (These are roughly the strongest size lower bounds we know for general  $\text{TC}^0$  circuits even in the worst case [11].) Using their ideas, they gave the first (zero-error randomized) improvement to brute-force-search for satisfiability algorithms (and indeed even #SAT algorithms) for constant depth  $\text{TC}^0$  circuits of size at most  $n^{1+\varepsilon_d}$ .

The lower bound results of [7] were extended to the much more powerful class of *k*-PTF circuits (of roughly the same size as [7]) by Kane, Kabanets and Lu [13]. In a follow-up paper, Kabanets and Lu [14] considered the satisfiability question for *k*-PTF circuits, and

<sup>8</sup> Note, crucially, that only the fan-in of a gate counts towards its size. So any gate computing a *k*-PTF on  $m$  variables only adds  $m$  to the size of the circuit, though of course the polynomial representing this PTF may have  $\approx m^k$  monomials.

<sup>9</sup>  $\text{ACC}^0 \circ \text{LTF}$  is a subclass of  $\text{TC}^0$  where general threshold gates are allowed only just above the variables. All computations above these gates are one of AND, OR or Modular gates (that count the number of inputs modulo a constant). It is suspected (but not proved) that subexponential-sized  $\text{ACC}^0$  circuits cannot simulate even a single general threshold gate. Hence, it is not clear if the class of subexponential-sized  $\text{ACC}^0 \circ \text{LTF}$  circuits contains even depth-2  $\text{TC}^0$  circuits of linear size.

<sup>10</sup> More specifically, the algorithm of Sakai et al. [22] works as long as the weight of the input polynomial  $P \in \mathbb{Z}[x_1, \dots, x_n]$  is bounded by  $\exp(n^{1-\Omega(1)})$  (or equivalently,  $M \leq O(n^{1-\Omega(1)})$ ).

could resolve this question in the special case that each PTF is subquadratically sparse, i.e. has  $n^{2-\Omega(1)}$  monomials. One of the reasons for this sparsity restriction is that their strategy does not seem to yield a SAT algorithm for a single degree-2 PTF (which is a depth-1 2-PTF circuit of *linear* size).

## 1.1 Proof outline

For simplicity we discuss SAT algorithms instead of #SAT algorithms.

### Satisfiability algorithm for $k$ -PTFs

At a high level, we follow the same strategy as Sakai et al. [22]. Their algorithm uses *memoization*, which is a standard and very useful strategy for satisfiability algorithms (see, e.g. [23]). Let  $\mathcal{C}$  be a circuit class and  $\mathcal{C}_n$  be the subclass of circuits from  $\mathcal{C}$  that have  $n$  variables. Memoization algorithms for  $\mathcal{C}$ -SAT fit into the following two-step template.

- Step 1: Solve by brute-force all instances of  $\mathcal{C}$ -SAT where the input circuit  $C' \in \mathcal{C}_m$  for some suitable  $m \ll n$ . (Typically,  $m = n^\varepsilon$  for some constant  $\varepsilon$ .) Usually this takes  $\exp(m^{O(1)}) \ll 2^n$  time.
- Step 2: On the input  $C \in \mathcal{C}_n$ , set all input variables  $x_{m+1}, \dots, x_n$  to Boolean values and for each such setting, obtain  $C'' \in \mathcal{C}_m$  on  $m$  variables. Typically  $C''$  is a circuit for which we have solved satisfiability in Step 1 and hence by a simple table lookup, we should be able to check if  $C''$  is satisfiable in  $\text{poly}(|C|)$  time. Overall, this takes time  $O^*(2^{n-m}) \ll 2^n$ .

At first sight, this seems perfect for  $k$ -PTFs, since it is a standard result that the number of  $k$ -PTFs on  $m$  variables is at most  $2^{O(m^{k+1})}$  [8]. Thus, Step 1 can be done in  $2^{O(m^{k+1})} \ll 2^n$  time.

For implementing Step 2, we need to ensure that the lookup (for satisfiability for  $k$ -PTFs on  $m$  variables) can be done quickly. Unfortunately how to do this is unclear. The following two ways suggest themselves.

- Store all polynomials  $P' \in \mathbb{Z}[x_1, \dots, x_m]$  with small coefficients. Since every  $k$ -PTF  $f$  can be sign-represented by an integer polynomial with coefficients of size  $2^{\text{poly}(m)}$  [17], this can be done with a table of size  $2^{\text{poly}(m)}$  and in time  $2^{\text{poly}(m)}$ . When the coefficients are small (say of bit-complexity  $\leq n^{o(1)}$ ), then this strategy already yields a #SAT algorithm, as observed by Sakai et al. [22]. Unfortunately, in general, given a restriction  $P'' \in \mathbb{Z}[x_1, \dots, x_m]$  of a polynomial  $P \in \mathbb{Z}[x_1, \dots, x_n]$ , its coefficients can be much larger (say  $2^{\text{poly}(n)}$ ) and it is not clear how to efficiently find a polynomial with small coefficients that sign-represents the same function.
- It is also known that every  $k$ -PTF on  $m$  variables can be uniquely identified by  $\text{poly}(m)$  numbers of bit-complexity  $O(m)$  each [8]: these are called the ‘‘Chow parameters’’ of  $f$ . Again for this representation, it is unclear how to compute efficiently the Chow parameters of the function represented by the restricted polynomial  $P''$ . (Even for an LTF, computing the Chow parameters is as hard as Subset-sum [19].)

The way we solve this problem is by using a beautiful recent result of Kane, Lovett, Moran and Zhang [15], who show that there is a simple decision tree that, when given as input the coefficients of any degree- $k$  polynomial  $P' \in \mathbb{Z}[x_1, \dots, x_m]$ , can determine the sign of the polynomial  $P'$  at all points in  $\{-1, 1\}^m$  using only  $\text{poly}(m)$  queries to the coefficients of  $P$ . Here, each query is a linear inequality on the coefficients of  $P$ ; such a decision tree is called a *linear decision tree*.

Our strategy is to replace Step 1 with the construction of this linear decision tree (which can be done in  $\exp(m^{O(1)})$  time). At each leaf of the linear decision tree, we replace the truth table of the input polynomial  $P'$  by a single bit that indicates whether  $f' = \text{sgn}(P')$  is satisfiable or not.

In Step 2, we simply run this decision tree on our restricted polynomial  $P''$  and obtain the answer to the corresponding satisfiability query in  $\text{poly}(m, w(P''))$  time. Note, crucially, that the height of the linear decision tree implied by [15] construction is  $\text{poly}(m)$  and *independent* of the bit-complexity of the coefficients of the polynomial  $P''$  (which may be as big as  $\text{poly}(n)$  in our algorithm). This concludes the description of the algorithm for  $k$ -PTF.

### Satisfiability algorithm for $k$ -PTF circuits

For  $k$ -PTF circuits, we follow a template set up by the result of Kabanets and Lu [14] on sparse-PTF circuits. We start by describing this template and then describe what is new in our algorithm.

The Kabanets-Lu algorithm is an induction on the depth  $d$  of the circuit (which is a fixed constant). Given as input a depth  $d$   $k$ -PTF circuit  $C$  on  $n$  variables, Kabanets and Lu do the following:

**Depth-reduction:** In [14], it is shown that on a random restriction that sets all *but*  $n^{1-2\beta}$  variables (here, think of  $\beta$  as a small constant, say 0.01) to random Boolean values, the bottom layer of  $C$  simplifies in the following sense.

All but  $t \leq n^\beta$  gates at the bottom layer become *exponentially* biased, i.e. on all but  $\delta = \exp(-n^{\Omega(1)})$  fraction of inputs they are equal to a fixed  $b \in \{-1, 1\}$ . Now, for each such biased gate  $g$ , there is a minority value  $b_g \in \{-1, 1\}$  that it takes on very few inputs. [14] show how to enumerate this small number of inputs in  $\delta \cdot 2^n$  time and check if there is a satisfying assignment among these inputs. Having ascertained that there is no such assignment, we replace these gates by their majority value and there are only  $t$  gates at the bottom layer. At this point, we “guess” the output of these  $t$  “unbiased” gates and for each such guess  $\sigma \in \{-1, 1\}^t$ , we check if there is an assignment that simultaneously satisfies:

- (a) the depth  $d - 1$  circuit  $C'$ , obtained by setting the unbiased gates to the guess  $\sigma$ , is satisfied.
- (b) each unbiased gate  $g_i$  evaluates to the corresponding value  $\sigma_i$ .

**Base case:** Continuing this way, we eventually get to a base case which is an AND of sparse PTFs for which there is a satisfiability algorithm using the polynomial method.

In the above algorithm, there are two steps where subquadratic sparsity is crucially used. The first is the minority assignment enumeration algorithm for PTFs, which uses ideas of Chen and Santhanam [6] to reduce the problem to enumerating biased LTFs, which is easy [7]. The second is the base case, which uses a non-trivial polynomial approximation for LTFs [24]. Neither of these results hold for even degree-2 PTFs in general. To overcome this, we do the following.

**Enumerating minority assignments.** Given a  $k$ -PTF on  $m$  variables that is  $\delta = \exp(-n^{\Omega(1)})$ -close to  $b \in \{-1, 1\}$ , we enumerate its minority assignments as follows. First, we set up a linear decision tree as in the  $k$ -PTF satisfiability algorithm. Then we set all but  $q \approx \log \frac{1}{\delta}$  variables of the PTF. On most such settings, the resulting PTF becomes the constant function and we can check this using the linear decision tree we created earlier. In this setting, there is nothing to do. Otherwise, we brute-force over the remaining variables to find the minority

assignments. Setting parameters suitably, this yields an  $O(\sqrt{\delta} \cdot 2^m)$  time algorithm to find the minority assignments of a  $\delta$ -biased  $k$ -PTF on  $m$  variables.

**Base case.** Here, we make the additional observation (which [14] do not need) that the AND of PTFs that is obtained further is *small* in that it only has slightly superlinear size. Hence, we can apply another random restriction in the style of [14] and using the minority assignment enumeration ideas, reduce it to an AND of a small (say  $n^{0.1}$ ) number of PTFs on  $n^{0.01}$  (say) variables. At this point, we can again run the linear decision tree (in a slightly more generalized form) to check satisfiability.

## 2 A result of Kane, Lovett, Moran, and Zhang [15]

► **Definition 7** (Coefficient vectors.). Fix any  $k, m \geq 1$ . There are exactly  $r = \sum_{i=0}^k \binom{m}{i}$  many multilinear monomials of degree at most  $k$ . Any multilinear polynomial  $P(x_1, \dots, x_m)$  can be identified with a list of the coefficients of its monomials in lexicographic order (say) and hence with some vector  $w \in \mathbb{R}^r$ . We call  $w$  the *coefficient vector* of  $P$  and use  $\text{coeff}_{m,k}(P)$  to denote this vector. When  $m, k$  are clear from context, we will simply use  $\text{coeff}(P)$  instead of  $\text{coeff}_{m,k}(P)$ .

► **Definition 8** (Linear Decision Trees). A *Linear Decision Tree* for a function  $f : \mathbb{R}^r \rightarrow S$  (for some set  $S$ ) is a decision tree where each internal node is labelled by a linear inequality of the form  $\sum_{i=1}^r w_i z_i \geq \theta$  (here  $z_1, \dots, z_n$  denote the input variables). Depending on the answer to this linear inequality, computation proceeds to the left or right child of this node, and this process continues until a leaf is reached, which is labelled with an element of  $S$  that is the output of  $f$  on the given input.

The following construction of linear decision trees due to Kane, Lovett, Moran and Zhang [15] will be crucial for us.

► **Theorem 9.** *There is a randomized algorithm, which on input a positive integer  $r$ , a subset  $H \subseteq \{-1, 1\}^r$ , and an error parameter  $\varepsilon$ , produces a (random) linear decision tree  $T$  of depth  $\Delta = O(r \log r \cdot \log(|H|/\varepsilon))$  that computes a (random) function  $F : \mathbb{R}^r \rightarrow \{-1, 1\}^{|H|} \cup \{?\}$  that has the following properties.*

1. Each linear query has coefficients in  $\{-2, -1, 0, 1, 2\}$ .
2. Given as input any  $w \in \mathbb{R}^r$  such that  $\langle w, a \rangle \neq 0$  for all  $a \in \{-1, 1\}^r$ ,  $F(w)$  is either the truth table of the LTF defined by  $w$  (with constant term 0) on inputs from  $H \subseteq \{-1, 1\}^r$ , or is equal to ?. Further, we have  $\Pr_F[F(w) = ?] \leq \varepsilon$ .

The randomized algorithm runs in time  $2^{O(\Delta)}$ .

► **Remark.** The last statement in the above theorem is not formally stated in [15] but can easily be inferred from their proof and a remark [15, Page 363] on the ‘‘Computational Complexity’’ of their procedure.<sup>11</sup>

We will need a generalization of this theorem for evaluating (tuples of)  $k$ -PTFs. However, it is a simple corollary of this theorem.

► **Corollary 10.** *Fix positive constants  $k$  and  $c$ . Let  $r = \sum_{i=0}^k \binom{m}{i} = \Theta(m^k)$  denote the number of coefficients in a degree- $k$  multilinear polynomial in  $m$  variables. There is a randomized algorithm which on input positive integers  $m$  and  $\ell \leq m^c$  produces a (random)*

<sup>11</sup> We also thank Daniel Kane (personal communication) for telling us about this.



linear decision tree  $T$  of depth  $\Delta = O(\ell \cdot m^{k+1} \log m)$  that computes a (random) function  $F : \mathbb{R}^{r-\ell} \rightarrow \mathbb{N} \cup \{?\}$  that has the following properties.

1. Each linear query has coefficients in  $\{-2, -1, 0, 1, 2\}$ .
2. Given as input any  $\ell$ -tuple of coefficient vectors  $\bar{w} = (\text{coeff}_{m,k}(P_1), \dots, \text{coeff}_{m,k}(P_\ell)) \in \mathbb{R}^{r-\ell}$  such that  $P_i(a) \neq 0$  for all  $a \in \{-1, 1\}^m$ ,  $F(\bar{w})$  is either the number of common satisfying assignments to all the  $k$ -PTFs on  $\{-1, 1\}^m$  sign-represented by  $P_1, \dots, P_\ell$ , or is equal to ?. Further, we have  $\Pr_F[F(\bar{w}) = ?] \leq (1/2)$ .

The randomized algorithm runs in time  $2^{O(\Delta)}$ .

**Proof.** For each  $b \in \{-1, 1\}^m$ , define  $\text{eval}_b \in \{-1, 1\}^r$  to be the vector of all evaluations of multilinear monomials of degree at most  $k$ , taken in lexicographic order, on the input  $b$ . Define  $H \subseteq \{-1, 1\}^r$  to be the set  $\{\text{eval}_b \mid b \in \{-1, 1\}^m\}$ . Clearly,  $|H| \leq 2^m$ . Further, note that given any polynomial  $P(x_1, \dots, x_m)$  of degree at most  $k$ , the truth table of the  $k$ -PTF sign-represented by  $P$  is given by the evaluation of the LTF represented by  $\text{coeff}(P)$  at the points in  $H$ . Our aim, therefore, is to evaluate the LTFs corresponding to  $\text{coeff}(P_1), \dots, \text{coeff}(P_\ell)$  at all the points in  $H$ .

For each  $i$ , we use the randomized algorithm from Theorem 9 to produce a decision tree  $T_i$  that evaluates the Boolean function  $f_i : \{-1, 1\}^m \rightarrow \{-1, 1\}$  sign-represented by  $P_i$  (or equivalently, evaluating the LTF corresponding to  $\text{coeff}(P_i)$  at all points in  $H$ ) with error  $\varepsilon = 1/2\ell$ . Note that  $T_i$  has depth  $O(m^k \log m \cdot \log(2^m/\ell)) = O(m^{k+1} \log m)$  as  $\ell \leq m^c$ . The final tree  $T$  is obtained by simply running  $T_1, \dots, T_\ell$  in order, which is of depth  $O(\ell m^{k+1} \log m)$ . The tree  $T$  outputs the number of common satisfying assignments to all the  $f_i$  if all the  $T_i$ s succeed and ? otherwise. Since each  $T_i$  outputs ? with probability at most  $1/2\ell$ , the tree  $T$  outputs ? with probability at most  $(1/2\ell) \cdot \ell = 1/2$ .

The claim about the running time follows from the analogous claim in Theorem 9 and the fact that the number of common satisfying assignments to all the  $f_i$  can be computed from the truth tables in  $2^{O(m)}$  time. This completes the proof.  $\blacktriangleleft$

### 3 The PTF-SAT algorithm

We are now ready to prove Theorem 3. We first state the algorithm, which follows a standard memoization idea (see, e.g. [23]). We assume that the input is a polynomial  $P \in \mathbb{Z}[x_1, \dots, x_n]$  of degree at most  $k$  that sign-represents a Boolean function  $f$  on  $n$  variables. The parameters of the instance are assumed to be  $(n, M)$ . Set  $m = n^{1/(k+1)} / \log n$ .

#### Algorithm $\mathcal{A}$ .

1. Use  $n_1 = 10n$  independent runs of the algorithm from Corollary 10 with  $\ell = 1$  to construct independent random linear decision trees  $T_1, \dots, T_{n_1}$  such that on any input polynomial  $Q(x_1, \dots, x_m)$  (or more precisely  $\text{coeff}_{m,k}(Q)$ ) of degree at most  $k$  that sign-represents a  $k$ -PTF  $g$  on  $m$  variables, each  $T_i$  computes the number of satisfying assignments to  $g$  with error at most  $1/2$ .
2. Set  $N = 0$ . ( $N$  will ultimately be the number of satisfying assignments to  $f$ .)
3. For each setting  $\sigma \in \{-1, 1\}^{n-m}$  to the variables  $x_{m+1}, \dots, x_n$ , do the following:
  - a. Compute the polynomial  $P_\sigma$  obtained by substituting the variables  $x_{m+1}, \dots, x_n$  accordingly in  $P$ .



- b. Run the decision trees  $T_1, \dots, T_{n_1}$  on  $\text{coeff}(P_\sigma)$  and compute their outputs. If all the outputs are ?, output ?. Otherwise, some  $T_i$  outputs  $N_\sigma$ , the number of satisfying assignments to  $P_\sigma$ . Add this to the current estimate to  $N$ .
4. Output  $N$ .

**Correctness.** It is clear from Corollary 10 and step 3b that algorithm  $\mathcal{A}$  outputs either ? or the correct number of satisfying assignments to  $f$ . Further, we claim that with probability at least  $1 - 1/2^{\Omega(n)}$ , the output is indeed the number of satisfying assignments to  $f$ . To see this, observe that it follows from Corollary 10 that for each setting  $\sigma \in \{-1, 1\}^{n-m}$  to the variables  $x_{m+1}, \dots, x_n$ , each linear decision tree  $T_i$  produced in step 1 errs on  $\text{coeff}(P_\sigma)$  (i.e. outputs ?) with probability at most  $1/2$ . The probability of each  $T_i$  doing so is thus at most  $1/2^{n_1}$ , as they are constructed independently. So the probability that the algorithm fails to determine  $N_\sigma$  is at most  $1/2^{n_1}$ . Finally, taking a union bound over all  $\sigma$ , which are  $2^{n-m}$  in number, we conclude that the probability of algorithm  $\mathcal{A}$  outputting ? is at most  $2^{n-m}/2^{n_1} \leq 1/2^{\Omega(n)}$ .

**Running time.** We show that the running time of algorithm  $\mathcal{A}$  is  $\text{poly}(n, M) \cdot 2^{n-m}$ . First note that by Corollary 10, the construction of a single linear decision tree  $T_i$  takes  $2^{O(\Gamma)}$  time, where  $\Gamma = m^{k+1} \log m$ , and hence, step 1 takes  $n_1 \cdot 2^{O(\Gamma)}$  time. Next, for a setting  $\sigma \in \{-1, 1\}^{n-m}$  to the variables  $x_{m+1}, \dots, x_n$ , computing  $P_\sigma$  and constructing the vector  $\text{coeff}(P_\sigma)$  takes only  $\text{poly}(n, M)$  time. Recall that the depth of each linear decision tree  $T_i$  is  $O(\Gamma)$  and thus, on input vector  $\text{coeff}(P_\sigma)$ , each of whose entries has bit complexity at most  $M$ , it takes time  $O(\Gamma) \cdot \text{poly}(M, n)$  to run all  $T_i$  and obtain the output  $N_\sigma$  or ?. Therefore, step 3 takes  $\text{poly}(n, M) \cdot 2^{n-m}$  time. Finally, the claim about the total running time of algorithm  $\mathcal{A}$  follows at once when we observe that for the setting  $m = n^{1/(k+1)}/\log n$ ,  $\Gamma = o(n/(\log n)^k) = o(n)$ .

## 4 Constant-depth circuits with PTF gates

In this section we give an algorithm for counting the number of satisfying assignment for a  $k$ -PTF circuit of constant depth and slightly superlinear size. We begin with some definitions.

► **Definition 11.** Let  $\delta \leq 1$  be any parameter. Two Boolean functions  $f, g$  are said to be  $\delta$ -close if  $\Pr_x[f(x) \neq g(x)] \leq \delta$ .

A  $k$ -PTF  $f$  specified by a polynomial  $P$  is said to be  $\delta$ -close to an explicit constant if it is  $\delta$ -close to a constant and such a constant can be computed efficiently, i.e.  $\text{poly}(n, M)$ , where  $n$  is the number of variables in  $P$  and  $w(P)$  is at most  $M$ .

► **Definition 12.** For a Boolean function  $f : \{-1, 1\}^n \rightarrow \{-1, 1\}$ , the majority value of  $f$  is the bit value  $b \in \{-1, 1\}$  which maximizes  $\Pr_x[f(x) = b]$  and the bit value  $-b$  is said to be its minority value.

For a Boolean function  $f$  with majority value  $b$ , an assignment  $x \in \{-1, 1\}^n$  is said to be a majority assignment if  $f(x) = b$  and minority assignment otherwise.

► **Definition 13.** Given a  $k$ -PTF  $f$  on  $n$  variables specified by a polynomial  $P$ , a parameter  $m \leq n$  and a partial assignment  $\sigma \in \{-1, 1\}^{n-m}$  on  $n-m$  variables, let  $P_\sigma$  be the polynomial obtained by substituting the variables in  $P$  according to  $\sigma$ . If  $P$  has parameters  $(n, M)$  then  $P_\sigma$  has parameters  $(m, M)$ . For a  $k$ -PTF circuit  $C$ ,  $C_\sigma$  is defined similarly. If  $C$  has parameters  $(n, s, d, M)$  then  $C_\sigma$  has parameters  $(m, s, d, M)$ .

**Outline of the #SAT procedure.** For designing a #SAT algorithm for  $k$ -PTF circuits, we use the generic framework developed by Kabanets and Lu [14] with some crucial modifications.

Given a  $k$ -PTF circuit  $C$  on  $n$  variables of depth  $d$  we want to count the number of satisfying assignments  $a \in \{-1, 1\}^n$  such that  $C(a) = -1$ . We in fact solve a slightly more general problem. Given  $(C, \mathcal{P})$ , where  $C$  is a small  $k$ -PTF circuit of depth  $d$  and  $\mathcal{P}$  is a set of  $k$ -PTF functions, such that  $\sum_{f \in \mathcal{P}} \text{fan-in}(f)$  is small, we count the number of assignments that simultaneously satisfy  $C$  and all the function in  $\mathcal{P}$ .

At the core of the algorithm that solves this problem, Algorithm  $\mathcal{B}$ , is a recursive procedure  $\mathcal{A}_5$ , which works as follows: on inputs  $(C, \mathcal{P})$  it first applies a simplification step that outputs  $\ll 2^n$  instances of the form  $(C', \mathcal{P}')$  such that

- Both  $C'$  and functions in  $\mathcal{P}'$  are on  $m \ll n$  variables.
- The sets of satisfying assignments of these instances “almost” partition the set of satisfying assignments of  $(C, \mathcal{P})$ .
- With all but very small probability the bottom layer of  $C'$  has the following nice structure.
  - At most  $n$  gates are  $\delta$ -biased. We denote this set of gates by  $B$  (as we will simplify them by setting them to the values they are biased towards).
  - At most  $n^{\beta d}$  gates are not  $\delta$ -biased. We denote these gates by  $G$  (as we will simplify them by “guessing” their values).
- There is a small set of satisfying assignments that are not covered by the satisfying assignments of  $(C', \mathcal{P}')$  but we can count these assignments with a brute-force algorithm that does not take too much time.

For each  $C'$  with this nice structure, then we try to use this structure to create  $C''$  which has depth  $d - 1$ . Suppose we reduce the depth as follows:

- Set all the gates in  $B$  to the values that they are biased towards.
- Try all the settings of the values that the gates in  $G$  can take, thereby from  $C'$  creating possibly  $2^{n^{\beta d}}$  instances  $(C'', \mathcal{P}')$ .

$(C'', \mathcal{P}')$  now is an instance where  $C''$  has depth  $d - 1$ . Unfortunately, by simply setting biased gates to the values they are biased towards, we may miss out on the minority assignments to these gates which could eventually satisfy  $C'$ . We design a subroutine  $\mathcal{A}_3$  to precisely handle this issue, i.e. to keep track of the number of minority assignments, say  $N_{C'}$ . This part of our algorithm is completely different from that of [14], which only works for subquadratically sparse PTFs.

Once  $\mathcal{A}_3$  has computed  $N_{C'}$ , i.e. the number of satisfying assignments among the minority assignments, we now need to only count the number of satisfying assignments among the rest of the assignments.

To achieve this we use an idea similar to that in [7, 14], which involves appending  $\mathcal{P}'$  with a few more  $k$ -PTFs (this forces the biased gates to their majority values). This gives say a set  $\tilde{\mathcal{P}}'$ . Similarly, while setting gates in  $G$  to their guessed values, we again use the same idea to ensure that we are counting satisfying assignments consistent with the guessed values, once again updating  $\tilde{\mathcal{P}}'$  to a new set  $\mathcal{P}''$ . This creates instances of the form  $(C'', \mathcal{P}'')$ , where the depth of  $C''$  is  $d - 1$ .

This way, we iteratively decrease the depth of the circuit by 1. Finally, we have instances  $(C'', \mathcal{P}'')$  such that the depth of  $C''$  is 1, i.e. it is a single  $k$ -PTF, say  $h$ . At this stage we solve #SAT( $\tilde{C}$ ), where  $\tilde{C} = h \wedge \bigwedge_{f \in \mathcal{P}''} f$ . This is handled in a subroutine  $\mathcal{A}_4$ . Here too our algorithm differs significantly from [14].

In what follows we will prove Theorem 6. In order to do so, we will set up various subroutines  $\mathcal{A}_1, \mathcal{A}_2, \mathcal{A}_3, \mathcal{A}_4, \mathcal{A}_5$  designed to accomplish certain tasks and combine them together at the end to finally design algorithm  $\mathcal{B}$  for the #SAT problem for  $k$ -PTF circuits.

$\mathcal{A}_1$  will be an oracle, used in other routines, which will compute number of common satisfying assignments for small AND of PTFs on few variables (using the same idea as in the algorithm for #SAT for  $k$ -PTFs).  $\mathcal{A}_2$  will be a simplification step, which will allow us to argue about some structure in the circuit (this algorithm is from [14]). It will make many gates at the bottom of the circuit  $\delta$ -close to a constant, thus simplifying it.  $\mathcal{A}_3$  will be used to count minority satisfying assignments for a bunch of  $\delta$ -biased PTFs, i.e. assignments which cause at least one of the PTFs to evaluate to its minority value.  $\mathcal{A}_4$  will be a general base of case of our algorithm, which will count satisfying assignments for AND of superlinear many PTFs, by first using  $\mathcal{A}_2$  to simplify the circuit, then reducing it to the case of small AND of PTFs and then using  $\mathcal{A}_1$ .  $\mathcal{A}_5$  will be a recursive procedure, which will use  $\mathcal{A}_2$  to first simplify the circuit, and then convert it into a circuit of lower depth, finally making a recursive call on the simplified circuit.

**Parameter setting.** Let  $d$  be a constant. Let  $A, B$  be two fixed absolute large constants. Let  $\zeta = \min(1, A/2Bk^2)$ . For each  $2 \leq i \leq d$ , let  $\beta_i = A \cdot \varepsilon_i$  and  $\varepsilon_i = (\frac{\zeta}{10A(k+1)})^i$ . Choose  $\beta_1 = 1/10$  and  $\varepsilon_1 = 1/10A$ .

**Oracle access to a subroutine.** Let  $\mathcal{A}_1(n', s, f_1, \dots, f_s)$  denote a subroutine with the following specification. Here,  $n$  is the number of variables in the original input circuit.

**Input:** AND of  $k$ -PTFs, say  $f_1, \dots, f_s$  specified by polynomials  $P_1, \dots, P_s$  respectively, such that  $s \leq n^{0.1}$  and for each  $i \in [s]$ ,  $f_i$  is defined over  $n' \leq n^{1/(2(k+1))}$  variables and  $w(P_i) \leq M$ .

**Output:**  $\#\{a \in \{-1, 1\}^{n'} \mid \forall i \in [s], P_i(a) = -1\}$ .

In what follows, we will assume that we have access to the above subroutine  $\mathcal{A}_1$ . We will set up such an oracle and show that it answers any call to it in time  $\text{poly}(n, M)$  in Section 4.5.

#### 4.1 Simplification of a $k$ -PTF circuit

For any  $1 > \varepsilon \gg (\log n)^{-1}$ , let  $\beta = A\varepsilon$  and  $\delta = \exp(-n^{\beta/B \cdot k^2})$ , where  $A$  and  $B$  are constants. Note that it is these constants  $A, B$  we use in the parameter settings paragraph above. Let  $\mathcal{A}_2(C, d, n, M)$  be the following subroutine.

**Input:**  $k$ -PTF circuit  $C$  of depth  $d$  on  $n$  variables with size  $n^{1+\varepsilon}$  and weight  $M$ .

**Output:** A decision tree  $T_{DT}$  of depth  $n - n^{1-2\beta}$  such that for a uniformly random leaf  $\sigma \in \{-1, 1\}^{n-n^{1-2\beta}}$  it outputs a *good* circuit  $C_\sigma$  with probability  $1 - \exp(-n^\varepsilon)$ , where  $C_\sigma$  is called good if its bottom layer has the following structure:

- there are at most  $n$  gates which are  $\delta$ -close to an explicit constant. Let  $B_\sigma$  denote this set of gates.
- there are at most  $n^\beta$  gates that are not  $\delta$ -close to an explicit constant. Let us denote this set of gates by  $G_\sigma$ .

In [14], such a subroutine  $\mathcal{A}_2(C, d, n, M)$  was designed. Specifically, they proved the following theorem.

► **Theorem 14** (Kabanets and Lu [14]). *There is a zero-error randomized algorithm  $\mathcal{A}_2(C, d, n, M)$  that runs in time  $\text{poly}(n, M) \cdot O(2^{n-n^{1-2\beta}})$  and outputs a decision tree as described above with probability at least  $1 - 1/2^{10n}$  (and outputs ? otherwise). Moreover, given a good  $C_\sigma$ , there is a deterministic algorithm that runs in time  $\text{poly}(n, M)$  which computes  $B_\sigma$  and  $G_\sigma$ .*

► **Remark.** In [14], it is easy to see that the probability of outputting ? is at most  $1/2$ . To bring down this probability to  $1/2^{10n}$ , we run their procedure in parallel  $10n$  times, and output the first tree that is output by the algorithm. The probability that no such tree is output is  $1/2^{10n}$ .

► **Remark.** In designing the above subroutine in [14], they consider a more general class of polynomially sparse-PTF circuits (i.e. each gate computes a PTF with polynomially many monomials) as opposed to the  $k$ -PTF circuits we consider here. Under this weaker assumption, they get that  $\delta = \exp(-n^{\Omega(\beta^3)})$ . However, by redoing their analysis for degree  $k$ -PTFs, it is easy to see that  $\delta$  could be set to  $\exp(-n^{\beta/B \cdot k^2})$  for some constant  $B$ . Under this setting of  $\delta$ , we get exactly the same guarantees. In this sense, the above theorem statement is a slight restatement of [14, Theorem 3.11].

## 4.2 Enumerating the minority assignments

We now design an algorithm  $\mathcal{A}_3(m, \ell, \delta, g_1, \dots, g_\ell)$ , which has the following behaviour.

**Input:** parameters  $m \leq n, \ell, \delta$  such that  $\delta \in [\exp(-m^{1/10(k+1)}), 1]$ ,  $\ell \leq m^2$ ,  $k$ -PTFs  $g_1, g_2, \dots, g_\ell$  specified by polynomials  $P_1, \dots, P_\ell$  on  $m$  variables  $(x_1, \dots, x_m)$  each of weight at most  $M$  and which are  $\delta$ -close to  $-1$ .

**Oracle access to:**  $\mathcal{A}_1$ .

**Output:**  $a \in \{-1, 1\}^m$  such that  $\exists i \in [\ell]$  for which  $P_i(a) > 0$ .

► **Lemma 15.** *There is a deterministic algorithm  $\mathcal{A}_3(m, \ell, \delta, g_1, \dots, g_\ell)$  as specified above that runs in time  $\text{poly}(m, M) \cdot \sqrt{\delta} \cdot 2^m$ .*

**Proof.** We start with the description of the algorithm.

$\mathcal{A}_3(m, \ell, \delta, g_1, \dots, g_\ell)$ .

1. Set  $q = \frac{1}{2} \log \frac{1}{\delta} \leq \frac{m}{2}$  and let  $\mathcal{N} = \emptyset$ . ( $\mathcal{N}$  will eventually be the collection of minority assignments i.e. all  $a \in \{-1, 1\}^m$  such that  $\exists i \in [\ell]$  for which  $P_i(a) > 0$ .)
2. For each setting  $\rho \in \{-1, 1\}^{m-q}$  to the variables  $x_{q+1}, \dots, x_m$ , do the following:
  - a. Construct the restricted polynomials  $P_{1,\rho}, \dots, P_{\ell,\rho}$ . Let  $g_{i,\rho} = \text{sgn}(P_{i,\rho})$  for  $i \in [\ell]$ .
  - b. Using oracle  $\mathcal{A}_1(q, 1, -g_{i,\rho})$ , check for each  $i \in [\ell]$  if  $g_{i,\rho}$  is the constant function  $-1$  by checking if the output of the oracle on the input  $-g_{i,\rho}$  is zero.
  - c. If there is an  $i \in [\ell]$  such that  $g_{i,\rho}$  is not the constant function  $-1$ , try all possible assignments  $\chi$  to the remaining  $q$  variables  $x_1, \dots, x_q$ . This way, enumerate all assignments  $b = (\chi, \rho)$  to  $x_1, \dots, x_m$  for which there is an  $i \in [\ell]$  such that  $P_i(b) > 0$ . Add such an assignment to the collection  $\mathcal{N}$ .
3. Output  $\mathcal{N}$ .

**Correctness.** If  $a \in \{-1, 1\}^m$  is a minority assignment (i.e.  $\exists i_0 \in [\ell]$  so that  $P_{i_0}(a) < 0$ ) and if  $a = (\chi, \rho)$  where  $\rho$  is an assignment to the last  $m - q$  variables, and  $\chi$  to the first  $q$ ,  $a$  will get added to  $\mathcal{N}$  in the loop of step 2 corresponding to  $\rho$  and that of  $\chi$  in step 2c, because of  $i_0$  being a witness. Conversely, observe that we only add to the collection  $\mathcal{N}$  when we encounter a minority assignment.

**Running time.** For each setting  $\rho \in \{-1, 1\}^{m-q}$  to the variables  $x_{q+1}, \dots, x_m$ , step 2a takes  $\text{poly}(m, M)$  time and step 2b takes  $O(\ell) = O(m^2)$  time and so combined, they take only  $\text{poly}(m, M)$  time. Let  $\mathcal{T}$  be the set consisting of all assignments  $\rho$  to the last  $m - q$  variables such that the algorithm enters the loop described in step 2c i.e.

$$\mathcal{T} = \{\rho \in \{-1, 1\}^{m-q} \mid \exists i \in [\ell] : g_{i,\rho} \text{ is not the constant function } -1\}$$

and let  $\mathcal{T}^c$  denote its complement. Also note that for a  $\rho \in \mathcal{T}$ , enumeration of minority assignments in step 2c takes  $2^q \cdot \ell \cdot \text{poly}(m, M)$  time. Therefore, we can bound the total running time by

$$\text{poly}(m, M)(2^q \cdot |\mathcal{T}| + |\mathcal{T}^c|).$$

Next, we claim that the size of  $\mathcal{T}$  is small:

► **Lemma 16.**  $|\mathcal{T}| \leq \ell \cdot \sqrt{\delta} \cdot 2^{m-q}$ .

**Proof.** We define for  $i \in [\ell]$ ,  $\mathcal{T}_i = \{\rho \in \{-1, 1\}^{m-q} \mid g_{i,\rho} \text{ is not the constant function } -1\}$ . By the union bound, it is sufficient to show that  $|\mathcal{T}_i| \leq \sqrt{\delta} \cdot 2^{m-q}$  for a fixed  $i \in [\ell]$ . Let  $D_m$  denote the uniform distribution on  $\{-1, 1\}^m$  i.e. on all possible assignments to the variables  $x_1, \dots, x_m$ . Then from the definition of  $\delta$ -closeness, we know

$$\Pr_{a \sim D_m} [g_i(a) = 1] \leq \delta.$$

Writing LHS in the following way, we have

$$\mathbf{E}_{\rho \sim D_{m-q}} \left[ \Pr_{\chi \sim D_q} [g_{i,\rho}(\chi) = 1] \right] \leq \delta$$

where  $D_{m-q}$  and  $D_q$  denote uniform distributions on assignments to the last  $m-q$  variables and the first  $q$  variables respectively. By Markov's inequality,

$$\Pr_{\rho \sim D_{m-q}} \left[ \Pr_{\chi \sim D_q} [g_{i,\rho}(\chi) = 1] \geq \sqrt{\delta} \right] \leq \sqrt{\delta}$$

Consider a  $\rho$  for which this event does not occur i.e. for which  $\Pr_{\chi \sim D_q} [g_{i,\rho}(\chi) = 1] < \sqrt{\delta}$ . For such a  $\rho$ ,  $g_{i,\rho}$  has only  $2^q = 1/\sqrt{\delta}$  many inputs and therefore,  $g_{i,\rho}$  must be the constant function  $-1$ . Thus, we conclude that

$$\Pr_{\rho \sim D_{m-q}} [g_{i,\rho} \text{ is not the constant function } -1] \leq \sqrt{\delta}$$

or in other words,  $|\mathcal{T}_i| \leq \sqrt{\delta} \cdot 2^{m-q}$ . ◀

Finally, by using the trivial bound  $|\mathcal{T}^c| \leq 2^{m-q}$  and the above claim, we obtain a total running time of  $\text{poly}(m, M) \cdot \sqrt{\delta} \cdot 2^m$  and this concludes the proof of the lemma. ◀

### 4.3 #SAT for AND of $k$ -PTFs

We design an algorithm  $\mathcal{A}_4(n, M, g_1, \dots, g_\tau)$  with the following functionality.

**Input:** A set of  $k$ -PTFs  $g_1, \dots, g_\tau$  specified by polynomials  $P_1, \dots, P_\tau$  on  $n$  variables such that  $w(p_i) \leq M$  for each  $i \in [\tau]$  and  $\sum_{i \in [\tau]} \text{fan-in}(g_i) \leq n^{1+\epsilon_1}$ .

**Output:**  $\#\{a \in \{-1, 1\}^n \mid \forall i \in [\tau], P_i(a) < 0\}$ .

### 4.3.1 The details of the algorithm

$\mathcal{A}_4(\mathbf{n}, \mathbf{M}, \mathbf{g}_1, \dots, \mathbf{g}_\tau)$ .

1. Let  $m = n^\alpha$  for  $\alpha = \frac{\zeta \varepsilon_1}{2(k+1)}$ . Let  $C$  denote the AND of  $g_1, \dots, g_\tau$ .
2. Run  $\mathcal{A}_2(C, 2, n, M)$  to obtain the decision tree  $T_{DT}$ . Initialize  $N$  to 0.
3. For each leaf  $\sigma$  of  $T_{DT}$ , do the following:
  - (A) If  $C_\sigma$  is not *good*, count the number of satisfying assignments for  $C_\sigma$  by brute-force and add to  $N$ .
  - (B) If  $C_\sigma$  is *good*, do the following:
    - (i)  $C_\sigma$  is now an AND of PTFs in  $B_\sigma$  and  $G_\sigma$ , over  $n' = n^{1-2\beta_1}$  variables, where all PTFs in  $B_\sigma$  are  $\delta$ -close to an explicit constant, where  $\delta = \exp(-n^{\beta_1/B \cdot k^2})$ . Moreover,  $|B_\sigma| \leq n, |G_\sigma| \leq n^{\beta_1}$ .  
Let  $B_\sigma = \{h_1, \dots, h_\ell\}$  be specified by  $Q_1, \dots, Q_\ell$ .  
Suppose for  $i \in [\ell]$ ,  $h_i$  is close to  $a_i \in \{-1, 1\}$ . Then let  $Q'_i = -a_i \cdot Q_i$  and  $h'_i = \text{sgn}(Q'_i)$ . Let  $B'_\sigma = \{Q'_1, \dots, Q'_\ell\}$ .
    - (ii) For each restriction  $\rho : \{x_{m+1}, \dots, x_{n'}\} \rightarrow \{-1, 1\}$ , do the following:
      - (a) Check if there exists  $h' \in B'_\sigma$  such that  $h'_\rho$  is not the constant function  $-1$  using  $\mathcal{A}_1(m, 1, h'_\rho)$ .
      - (b) If such an  $h' \in B'_\sigma$  exists, then count the number of satisfying assignments for  $C_{\sigma\rho}$  by brute-force and add to  $N$ .
      - (c) If the above does not hold, we have established that for each  $h_i \in B_\sigma$ ,  $h_{i,\rho}$  is the constant function  $a_i$ . If  $\exists i \in [\ell]$  such that  $a_i = 1$ , it means  $C_{\sigma\rho}$  is also a constant 1. Then simply halt. Else set each  $h_i$  to  $a_i$ .  
Thus,  $C_{\sigma\rho}$  has been reduced to an AND of  $n^{\beta_1}$  many PTFs over  $m$  variables. Call this set  $G'_{\sigma\rho}$ , use  $\mathcal{A}_1(m, n^{\beta_1}, G'_{\sigma\rho})$  to calculate the number of satisfying assignments and add the output to  $N$ .
4. Finally, output  $N$ .

### 4.3.2 The correctness argument and running time analysis

► **Lemma 17.**  $\mathcal{A}_4$  is a zero-error randomized algorithm that counts the number of satisfying assignments correctly. Further,  $\mathcal{A}_4$  runs in time  $\text{poly}(n, M) \cdot 2^{n-n^\alpha}$  and outputs the right answer with probability at least  $1/2$  (and outputs ? otherwise).

**Proof.**

**Correctness.** For a leaf  $\sigma$  of  $T_{DT}$ , when  $C_\sigma$  is not *good*, we simply use brute-force, which is guaranteed to be correct. Otherwise,

- If  $h'_\rho$  not the constant function  $-1$  for some  $h' \in B'_\sigma$ , then we again use brute-force, which is guaranteed to work correctly.
- Otherwise, for each  $h' \in B'_\sigma$ ,  $h'_\rho$  is the constant function  $-1$ . Here we only need to consider the satisfying assignments for the gates in  $G_{\sigma\rho}$ . For this we use  $\mathcal{A}_1$ , that works correctly by assumption.

Further, we need to ensure that the parameters that we call  $\mathcal{A}_1$  on, are valid. To see this, observe that  $m = n^\alpha \leq n^{1/(2(k+1))}$  because of the setting of  $\alpha$  and further, we have  $n^{\beta_1} \leq n^{0.1}$ .

Finally, the claim about the error probability follows from the error probability of  $\mathcal{A}_2$  (Theorem 14).

**Running Time.** The time taken for constructing  $T_{DT}$  is  $O^*(2^{n-n^{1-2\beta_1}})$ , by Theorem 14. For a leaf  $\sigma$  of  $T_{DT}$ , we know that step 3A is executed with probability at most  $2^{-n^{\varepsilon_1}}$ . The total time for running step 3A is thus  $O^*(2^{n-n^{\varepsilon_1}})$ . We know that the oracle  $\mathcal{A}_1$  answers

calls in  $\text{poly}(n, M)$  time. Hence, the total time for running step 3(B)iiia is  $O^*(2^{n-n^\alpha})$ . Next, note that if step 3(B)iiib is executed, then all PTFs in  $B_\sigma$  are  $\delta$ -close to  $-1$ . So, the number of times it runs is at most  $\delta \cdot 2^{n^\ell}$ . Therefore, the total time for running step 3(B)iiib is  $O^*(2^{n+n^\alpha-n^{\beta_1/Bk^2}})$ . Similar to the analysis of step 3(B)iiia, the total time for running step 3(B)iiic is also  $O^*(2^{n-n^\alpha})$ .

Summing them up, we conclude that total running time is  $O^*(2^{n-n^\alpha})$ , as due to our choice of various parameters,  $n - n^\alpha$  is the dominating power of 2. This completes the proof.  $\blacktriangleleft$

#### 4.4 #SAT for larger depth $k$ -PTF circuits

Let  $C$  be a  $k$ -PTF circuit of depth  $d \geq 1$  on  $n$  variables and let  $\mathcal{P}$  be a set of  $k$ -PTFs  $g_1, \dots, g_\tau$ , which are specified by  $n$ -variate polynomials  $P_1, \dots, P_\tau$ . Let  $\#\text{SAT}(C, \mathcal{P})$  denote  $\#\{a \in \{-1, 1\}^n \mid C(a) < 0 \text{ and } \forall i \in [\tau], P_i(a) < 0\}$ . We now specify our depth-reduction algorithm  $\mathcal{A}_5(n, d, M, n^{1+\varepsilon_d}, C, \mathcal{P})$ .

**Input:**  $(C, \mathcal{P})$  as follows:

- $k$ -PTF circuit  $C$  with parameters  $(n, n^{1+\varepsilon_d}, d, M)$ .
- a set  $\mathcal{P}$  of  $k$ -PTFs  $g_1, \dots, g_\tau$  on  $n$  variables, which are specified by polynomials  $P_1, \dots, P_\tau$  such that  $\sum_{i=1}^\tau \text{fan-in}(g_i) \leq n^{1+\varepsilon_d}$  and for each  $i \in [\tau]$ ,  $w(P_i) \leq M$ .

**Oracle access to:**  $\mathcal{A}_1, \mathcal{A}_4$ .

**Output:**  $\#\text{SAT}(C, \mathcal{P})$ .

We start by describing the algorithm.

##### 4.4.1 The details of the algorithm

Let `count` be a global counter initialized to 0 before the execution of the algorithm.

**$\mathcal{A}_5(n, d, M, n^{1+\varepsilon_d}, C, \mathcal{P})$ .**

1. If  $d = 1$ , output  $\mathcal{A}_4(n, M, \{C\} \cup \mathcal{P})$  and halt.
2. Run  $\mathcal{A}_2(C, d, n, M)$ , which gives us a  $\text{T}_{\text{DT}}$ .
3. For each leaf  $\sigma \in \{-1, 1\}^{n-n^{1-2\beta_d}}$  of  $\text{T}_{\text{DT}}$ . (If not, output ?.)
  - a. For each  $i \in [\tau]$  compute  $P_{i,\sigma}$ , the polynomial obtained by substituting  $\sigma$  in its variables. Let  $\mathcal{P}_\sigma = \{P_{1,\sigma}, \dots, P_{\tau,\sigma}\}$ .
  - b. Obtain  $C_\sigma$ . If  $C_\sigma$  is not a good circuit, then brute-force to find the number of satisfying assignments of  $(C_\sigma, \mathcal{P}_\sigma)$ , say  $N_\sigma$ , and set `count` = `count` +  $N_\sigma$ .
  - c. If  $C_\sigma$  is good then obtain  $B_\sigma$  and  $G_\sigma$ .
  - d. Let  $B_\sigma = \{h_1, \dots, h_\ell\}$  be specified by  $Q_1, \dots, Q_\ell$ . We know that each  $h \in B_\sigma$  is  $\delta$ -close to an explicit constant, for  $\delta = \exp(-n^{\beta_d/Bk^2})$ . Suppose for  $i \in [\ell]$ ,  $h_i$  is close to  $a_i \in \{-1, 1\}$ . Then let  $Q'_i = -a_i \cdot Q_i$  and  $h'_i = \text{sgn}(Q'_i)$ . Let  $B'_\sigma = \{Q'_1, \dots, Q'_\ell\}$ .
  - e. Run  $\mathcal{A}_3(n^{1-2\beta_d}, \ell, \delta, h'_1, \dots, h'_\ell)$  to obtain the set  $\mathcal{N}_\sigma$  of all the minority assignments of  $B_\sigma$ . (Note that this uses oracle access to  $\mathcal{A}_1$ .)  
for each  $a \in \mathcal{N}_\sigma$ , if  $((C(a) < 0) \text{ AND } (\forall i \in [\ell], P_i(a) < 0))$ , then `count` = `count` + 1.
  - f. Let  $G_\sigma = \{f_1, \dots, f_t\}$  be specified by polynomials  $R_1, \dots, R_t$ . We know that  $t \leq n^{\beta_d}$ . For each  $b \in \{-1, 1\}^t$ ,
    - i. Let  $R'_i = -b_i \cdot R_i$  for  $i \in [t]$ . Let  $G'_{\sigma,b} = \{R'_1, \dots, R'_t\}$ .
    - ii. Let  $C_{\sigma,b}$  be the circuit obtained from  $C_\sigma$  by replacing each  $h_i$  by  $a_i$   $1 \leq i \leq \ell$  and each  $f_j$  by  $b_j$  for  $1 \leq j \leq t$ .

- iii.  $\mathcal{P}_{\sigma,b} = \mathcal{P}_{\sigma} \cup B'_{\sigma} \cup G'_{\sigma,b}$ .
- iv. If  $d > 2$  then run  $\mathcal{A}_5(n^{1-2\beta_d}, d-1, M, n^{1+\varepsilon_d}, C_{\sigma,b}, \mathcal{P}_{\sigma,b})$   $n_1 = 10n$  times and let  $N_{\sigma}$  be the output of the first run that does not output ?. Set  $\text{count} = \text{count} + N_{\sigma}$ . (If all runs of  $\mathcal{A}_5$  output ?, then output ?.)
- v. If  $d = 2$  then run  $\mathcal{A}_4(n^{1-2\beta_d}, M, C_{\sigma,b} \cup \mathcal{P}_{\sigma,b})$   $n_1 = 10n$  times and let  $N_{\sigma}$  be the output of the first run that does not output ?. Set  $\text{count} = \text{count} + N_{\sigma}$ . (If all runs of  $\mathcal{A}_5$  output ?, then output ?.)

4. Output  $\text{count}$ .

#### 4.4.2 The correctness argument and running time analysis

► **Lemma 18.** *The algorithm  $\mathcal{A}_5$  described above is a zero-error randomized algorithm which on input  $(C, \mathcal{P})$  as described above, correctly  $\#SAT(C, \mathcal{P})$ . Moreover, the algorithm outputs the correct answer (and not ?) with probability at least  $1/2$ . Finally,  $\mathcal{A}_5(n, d, M, n^{1+\varepsilon_d}, C, \emptyset)$  runs in time  $\text{poly}(n, M) \cdot 2^{n - n^{\zeta\varepsilon_d/2(k+1)}}$ , where parameters  $\varepsilon_d, \zeta$  are as defined at the beginning of Section 4.*

**Proof.** We argue correctness by induction on the depth  $d$  of the circuit  $C$ .

Clearly, if  $d = 1$ , correctness follows from the correctness of algorithm  $\mathcal{A}_4$ . This takes care of the base case.

If  $d \geq 2$ , we argue first that if the algorithm does not output ?, then it does output  $\#SAT(C, \mathcal{P})$  correctly. Assume that the algorithm  $\mathcal{A}_2$  outputs a decision tree  $T_{DT}$  as required (otherwise, the algorithm outputs ? and we are done). Now, it is sufficient to argue that for each  $\sigma$ , the number of satisfying assignments to  $(C_{\sigma}, \mathcal{P}_{\sigma})$  is computed correctly (if the algorithm does not output ?).

Fix any  $\sigma$ . If  $C_{\sigma}$  is not a good circuit, then the algorithm uses brute-force to compute  $\#SAT(C_{\sigma}, \mathcal{P}_{\sigma})$  which yields the right answer. So we may assume that  $C_{\sigma}$  is indeed good.

Now, the satisfying assignments to  $(C_{\sigma}, \mathcal{P}_{\sigma})$  break into two kinds: those that are minority assignments to the set  $B_{\sigma}$  and those that are majority assignments to  $B_{\sigma}$ . The former set is enumerated in Step 3e (correctly by our analysis of  $\mathcal{A}_3$ ) and hence we count all these assignments in this step.

Finally, we claim that the satisfying assignments to  $(C_{\sigma}, \mathcal{P}_{\sigma})$  that are majority assignments of all gates in  $B_{\sigma}$  are counted in Step 3f. To see this, note that each such assignment  $a \in \{-1, 1\}^{n^{1-2\beta_d}}$  forces the gates in  $G_{\sigma}$  to some values  $b_1, \dots, b_t \in \{-1, 1\}$ . Note that for each such  $b \in \{-1, 1\}^t$ , these assignments are exactly the satisfying assignments of the pair  $(C_{\sigma,b}, \mathcal{P}_{\sigma,b})$  as defined in the algorithm. In particular, the number satisfying assignments to  $(C_{\sigma}, \mathcal{P}_{\sigma})$  that are majority assignments of all gates in  $B_{\sigma}$  can be written as

$$\sum_{b \in \{-1, 1\}^t} \#SAT(C_{\sigma,b}, \mathcal{P}_{\sigma,b}).$$

We now want to apply the induction hypothesis to argue that all the terms in the sum are computed correctly. To do this, we need to argue that the size of  $C_{\sigma,b}$  and the total fan-in of the gates in  $\mathcal{P}_{\sigma,b}$  are bounded as required (note that the total size of  $C$  remains the same, while the total fan-in of  $\mathcal{P}$  increases by the total fan-in of the gates in  $B'_{\sigma} \cup G'_{\sigma,b}$  which is at most  $n^{1+\varepsilon_d}$ ). It can be checked that this boils down to the following two inequalities

$$n^{(1-2\beta_d)(1+\varepsilon_d-1)} \geq n^{1+\varepsilon_d} \quad \text{and} \quad n^{(1-2\beta_d)(1+\varepsilon_d-1)} \leq 2n^{1+\varepsilon_d}$$

both of which are easily verified for our choice of parameters (for large enough  $n$ ). Thus, by the induction hypothesis, all the terms in the sum are computed correctly (unless we get ?). Hence, the output of the algorithm is correct by induction.



Now, we analyze the probability of error. If  $d = 1$ , the probability of error is at most  $1/2$  by the analysis of  $\mathcal{A}_4$ . If  $d > 2$ , we get an error if either  $\mathcal{A}_2$  outputs  $?$  or there is some  $\sigma$  such that the corresponding runs of  $\mathcal{A}_5$  or  $\mathcal{A}_4$  output  $?$ . The probability of each is at most  $1/2^{10n}$ . Taking a union bound over at most  $2^n$  many  $\sigma$ , we see that the probability of error is at most  $1/2^{\Omega(n)} \leq 1/2$ .

Finally, we analyze the running time. Define  $\mathcal{T}(n, d, M)$  to be the running time of the algorithm on a pair  $(C, \mathcal{P})$  as specified in the input description above. We need the following claim.

► **Lemma 19.**  $\mathcal{T}(n, d, M) \leq \text{poly}(n, M) \cdot 2^{n-n^{\zeta \varepsilon_d/2(k+1)}}$ .

To see the above, we argue by induction. The case  $d = 1$  follows from the running time of  $\mathcal{A}_4$ . Further from the description of the algorithm, we get the following inequality for  $d \geq 2$ .

$$\mathcal{T}(n, d, M) \leq \text{poly}(n, M) \cdot (2^{n-n^{1-2\beta_d}} + 2^{n-n^{\varepsilon_d}} + 2^{n-\frac{1}{2} \cdot n^{-\beta_d/(Bk^2)}} + 2^{n-n^{(1-2\beta_d)\zeta \varepsilon_d-1/2(k+1)}}) \quad (1)$$

The first term above accounts for the running time of  $\mathcal{A}_2$  and all steps other Steps 3b, 3e and 3f. The second term accounts for the brute force search in Step 3b since there are only a  $2^{-n^{\varepsilon_d}}$  fraction of  $\sigma$  where it is performed. The third term accounts for the minority enumeration algorithm in Step 3e (running time follows from the running time of that algorithm). The last term is the running time of Step 3f and follows from the induction hypothesis.

It suffices to argue that each term in the RHS of (1) can be bounded by  $2^{n-n^{\zeta \varepsilon_d/2(k+1)}}$ . This is an easy verification from our choice of parameters and left to the reader. This concludes the proof. ◀

## 4.5 Putting it together

In this subsection, we complete the proof of Theorem 6 using the aforementioned subroutines. We also need to describe the subroutine  $\mathcal{A}_1$ , which is critical for all the other subroutines. We shall do so *inside* our final algorithm for the #SAT problem for  $k$ -PTF circuits, algorithm  $\mathcal{B}$ . Recall that  $\mathcal{A}_1$  has the following specifications:

**Input:** AND of  $k$ -PTFs, say  $f_1, \dots, f_s$  specified by polynomials  $P_1, \dots, P_s$  respectively, such that  $s \leq n^{0.1}$  and for each  $i \in [s]$ ,  $f_i$  is defined over  $n' \leq n^{1/(2(k+1))}$  variables and  $w(P_i) \leq M$ .

**Output:**  $\#\{a \in \{-1, 1\}^{n'} \mid \forall i \in [s], f_i(a) = -1\}$ .

We are now ready to complete the proof of Theorem 6. Suppose  $C$  is the input  $k$ -PTF circuit with parameters  $(n, n^{1+\varepsilon_d}, d, M)$ . On these input parameters  $(C, n, n^{1+\varepsilon_d}, d, k, M)$ , we finally have the following algorithm for the #SAT problem for  $k$ -PTF circuits:

**$\mathcal{B}(C, n, n^{1+\varepsilon_d}, d, k, M)$ .**

1. (*Oracle Construction Step*) Construct the oracle  $\mathcal{A}_1$  as follows. Use  $n_1 = 10n$  independent runs of the algorithm from Corollary 10, with  $\ell$  chosen to be  $n^{0.1}$  and  $m$  to be  $n^{1/2(k+1)}$ , to construct independent random linear decision trees  $T_1, \dots, T_{n_1}$  such that on any input  $\bar{w} = (\text{coeff}_{m,k}(Q_1), \dots, \text{coeff}_{m,k}(Q_\ell)) \in \mathbb{R}^{r \cdot \ell}$  (where  $Q_i$ s are polynomials of degree at most  $k$  that sign-represent  $k$ -PTFs  $g_i$ , each on  $m$  variables), each  $T_i$  computes the number of common satisfying assignments to  $g_1, \dots, g_\ell$  with error at most  $1/2$ .
2. Run  $\mathcal{A}_5(n, d, M, n^{1+\varepsilon_d}, C, \emptyset)$ . For an internal call to  $\mathcal{A}_1$ , say on parameters  $(n', s, f_1, \dots, f_s)$  where  $n' \leq m$  and  $s \leq \ell$ , do the following:

- a. Run each  $T_i$  on the input  $\bar{w} = (\text{coeff}_{n',k}(P_1), \dots, \text{coeff}_{n',k}(P_s)) \in \mathbb{R}^{r \cdot s}$ . (We expand out the coefficient vectors with dummy variables so that they depend on exactly  $m$  variables. Similarly, using some dummy polynomials, we can assume that there are exactly  $\ell$  polynomials.)
- b. If some  $T_i$  outputs the number of common satisfying assignments to  $f_1, \dots, f_s$ , then output that. Otherwise, if all  $T_i$  output ?, then output ?.

► **Lemma 20.** *The construction of the zero-error randomized oracle  $\mathcal{A}_1$  in the above algorithm takes  $2^{O(n^{0.6})}$  time. Once constructed, the oracle  $\mathcal{A}_1$  answers any call (with the correct parameters) in  $\text{poly}(n, M)$  time with error at most  $1/2^{10n}$ .*

**Proof.**

**Correctness.** It is clear from Corollary 10 that algorithm  $\mathcal{A}_1$  outputs either ? or the correct number of common satisfying assignments to  $f_1, \dots, f_s$ . Further, as the  $T_i$ s in step 1 are constructed independently, it follows that with probability at least  $1 - 1/2^{10n}$ , the algorithm indeed outputs the number of common satisfying assignments to  $f_1, \dots, f_s$ .

**Running Time.** Substituting the parameters  $\ell = n^{0.1}$  and  $m = n^{1/(2(k+1))}$  in Corollary 10, we see that the construction of  $\mathcal{A}_1$  (step 1) takes  $n_1 \cdot 2^{n^{0.6}}$  time. Also, the claimed running time of answering a call follows upon observing that steps 2a and 2b combined take only  $\text{poly}(n, M)$  time to execute. ◀

With the correctness of  $\mathcal{A}_1$  now firmly established, we finally argue the correctness and running time of algorithm  $\mathcal{B}$ .

**Correctness.** The correctness of  $\mathcal{B}$  follows from that of  $\mathcal{A}_1, \mathcal{A}_2, \mathcal{A}_3, \mathcal{A}_4$ , and  $\mathcal{A}_5$  (see Lemma 20, Theorem 14, Lemmas 15, 17, and 18 respectively). Furthermore, if the algorithm  $\mathcal{A}_1$  is assumed to have no error at all, then from the analysis of  $\mathcal{A}_5$ , we see that the probability of error in  $\mathcal{B}$  is at most  $1/2$ . However, as algorithm  $\mathcal{A}_1$  is itself randomized, we still need to bound the probability that any of the calls made to  $\mathcal{A}_1$  produce an undesirable output (i.e. an output of ?). To this end, first note that as the running time of  $\mathcal{A}_5$  is bounded by  $2^n$ , the number of calls to  $\mathcal{A}_1$  is also bounded by  $2^n$ . But by Theorem 14 and Lemma 20, the probability of  $\mathcal{A}_1$  outputting ? is bounded by  $1/2^{10n}$ . Therefore, by the union bound, algorithm  $\mathcal{B}$  correctly outputs the number of satisfying assignments to the input circuit  $C$  with probability at least  $1/2 - 1/2^{\Omega(n)} \geq 1/4$ .

**Running Time.** By Lemma 18 and 20, the running time of  $\mathcal{B}$  will be  $2^{O(n^{0.6})} + \text{poly}(n, M) \cdot 2^{n - n^{\zeta \varepsilon_d / 2(k+1)}}$ . Thus, the final running time is  $\text{poly}(n, M) \cdot 2^{n-S}$  where  $S = n^{\zeta \varepsilon_d / 2(k+1)}$  and where  $\varepsilon_d > 0$  is a constant depending only on  $k$  and  $d$ . Setting  $\varepsilon_{k,d} = \zeta \varepsilon_d / 2(k+1)$  gives the statement of Theorem 6.

---

## References

- 1 Amir Abboud and Karl Bringmann. Tighter Connections Between Formula-SAT and Shaving Logs. In *45th International Colloquium on Automata, Languages, and Programming, ICALP 2018, July 9-13, 2018, Prague, Czech Republic*, pages 8:1–8:18, 2018.
- 2 Amir Abboud, Thomas Dueholm Hansen, Virginia Vassilevska Williams, and Ryan Williams. Simulating branching programs with edit distance and friends: or: a polylog shaved is a lower bound made. In *Proceedings of the 48th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2016, Cambridge, MA, USA, June 18-21, 2016*, pages 375–388, 2016.

- 3 Amir Abboud and Aviad Rubinfeld. Fast and Deterministic Constant Factor Approximation Algorithms for LCS Imply New Circuit Lower Bounds. In *9th Innovations in Theoretical Computer Science Conference, ITCS 2018, January 11-14, 2018, Cambridge, MA, USA*, pages 35:1–35:14, 2018.
- 4 Paul Beame, Stephen A. Cook, and H. James Hoover. Log Depth Circuits for Division and Related Problems. *SIAM J. Comput.*, 15(4):994–1003, 1986.
- 5 Lijie Chen. Toward Super-Polynomial Size Lower Bounds for Depth-Two Threshold Circuits. *CoRR*, abs/1805.10698, 2018. URL: <http://arxiv.org/abs/1805.10698>.
- 6 Ruiwen Chen and Rahul Santhanam. Improved algorithms for sparse MAX-SAT and MAX-k-CSP. In *International Conference on Theory and Applications of Satisfiability Testing*, pages 33–45. Springer, 2015.
- 7 Ruiwen Chen, Rahul Santhanam, and Srikanth Srinivasan. Average-Case Lower Bounds and Satisfiability Algorithms for Small Threshold Circuits. *Theory of Computing*, 14(9):1–55, 2018. doi:10.4086/toc.2018.v014a009.
- 8 Chao-Kong Chow. On the characterization of threshold functions. In *Switching Circuit Theory and Logical Design, 1961. SWCT 1961. Proceedings of the Second Annual Symposium on*, pages 34–38. IEEE, 1961.
- 9 William Hesse, Eric Allender, and David A. Mix Barrington. Uniform constant-depth threshold circuits for division and iterated multiplication. *J. Comput. Syst. Sci.*, 65(4):695–716, 2002.
- 10 Russell Impagliazzo, Shachar Lovett, Ramamohan Paturi, and Stefan Schneider. 0-1 Integer Linear Programming with a Linear Number of Constraints. *Electronic Colloquium on Computational Complexity (ECCC)*, 21:24, 2014.
- 11 Russell Impagliazzo, Ramamohan Paturi, and Michael E. Saks. Size-Depth Tradeoffs for Threshold Circuits. *SIAM J. Comput.*, 26(3):693–707, 1997.
- 12 Russell Impagliazzo, Ramamohan Paturi, and Stefan Schneider. A satisfiability algorithm for sparse depth two threshold circuits. In *Foundations of Computer Science (FOCS), 2013 IEEE 54th Annual Symposium on*, pages 479–488. IEEE, 2013.
- 13 Valentine Kabanets, Daniel M Kane, and Zhenjian Lu. A polynomial restriction lemma with applications. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing*, pages 615–628. ACM, 2017.
- 14 Valentine Kabanets and Zhenjian Lu. Satisfiability and Derandomization for Small Polynomial Threshold Circuits. In *LIPICs-Leibniz International Proceedings in Informatics*, volume 116, 2018.
- 15 Daniel M Kane, Shachar Lovett, Shay Moran, and Jiapeng Zhang. Active classification with comparison queries. In *Foundations of Computer Science (FOCS), 2017 IEEE 58th Annual Symposium on*, pages 355–366. IEEE, 2017.
- 16 Daniel M. Kane and Ryan Williams. Super-linear gate and super-quadratic wire lower bounds for depth-two and depth-three threshold circuits. In *Proceedings of the 48th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2016.*, pages 633–643, 2016.
- 17 S Muroga. Threshold logic and its applications. 1971.
- 18 Ryan O’Donnell. *Analysis of boolean functions*. Cambridge University Press, 2014.
- 19 Ryan O’Donnell and Rocco A. Servedio. The Chow Parameters Problem. *SIAM J. Comput.*, 40(1):165–199, 2011.
- 20 Ramamohan Paturi, Pavel Pudlák, Michael E Saks, and Francis Zane. An improved exponential-time algorithm for k-SAT. *Journal of the ACM (JACM)*, 52(3):337–364, 2005.
- 21 Ramamohan Paturi, Pavel Pudlák, and Francis Zane. Satisfiability coding lemma. In *Foundations of Computer Science, 1997. Proceedings., 38th Annual Symposium on*, pages 566–574. IEEE, 1997.

- 22 Takayuki Sakai, Kazuhisa Seto, Suguru Tamaki, and Junichi Teruyama. Bounded Depth Circuits with Weighted Symmetric Gates: Satisfiability, Lower Bounds and Compression. In *41st International Symposium on Mathematical Foundations of Computer Science (MFCS 2016)*, volume 58, pages 82:1–82:16, 2016.
- 23 Rahul Santhanam. Fighting Perebor: New and Improved Algorithms for Formula and QBF Satisfiability. In *51th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2010, October 23-26, 2010*, pages 183–192, 2010.
- 24 Srikanth Srinivasan. On improved degree lower bounds for polynomial approximation. In *LIPICs-Leibniz International Proceedings in Informatics*, volume 24. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2013.
- 25 Ryan Williams. A New Algorithm for Optimal Constraint Satisfaction and Its Implications. In *Automata, Languages and Programming: 31st International Colloquium, ICALP 2004, Turku, Finland, July 12-16, 2004. Proceedings*, pages 1227–1237, 2004.
- 26 Ryan Williams. Improving exhaustive search implies superpolynomial lower bounds. In *Proceedings of the forty-second ACM symposium on Theory of computing*, pages 231–240. ACM, 2010.
- 27 Ryan Williams. Non-uniform ACC Circuit Lower Bounds. In *Computational Complexity (CCC), 2011 IEEE 26th Annual Conference on*, pages 115–125. IEEE, 2011.
- 28 Ryan Williams. Faster all-pairs shortest paths via circuit complexity. In *Symposium on Theory of Computing, STOC 2014, New York, NY, USA, May 31 - June 03, 2014*, pages 664–673, 2014.
- 29 Ryan Williams. New algorithms and lower bounds for circuits with linear threshold gates. In *Proceedings of the forty-sixth annual ACM symposium on Theory of computing*, pages 194–202. ACM, 2014.
- 30 Virginia Vassilevska Williams. Hardness of Easy Problems: Basing Hardness on Popular Conjectures such as the Strong Exponential Time Hypothesis (Invited Talk). In *10th International Symposium on Parameterized and Exact Computation, IPEC 2015.*, pages 17–29, 2015.

# Small-Set Expansion in Shortcode Graph and the 2-to-2 Conjecture

**Boaz Barak**<sup>1</sup>

Harvard University School of Engineering and Applied Sciences, Cambridge, USA  
b@boazbarak.org

**Pravesh K. Kothari**<sup>2</sup>

Princeton University and IAS Princeton, USA  
kothari@cs.princeton.edu

**David Steurer**

ETH Zurich, Zurich, Switzerland  
dsteurer@inf.ethz.ch

---

## Abstract

Dinur, Khot, Kindler, Minzer and Safra (2016) recently showed that the (imperfect completeness variant of) Khot’s 2 to 2 games conjecture follows from a combinatorial hypothesis about the soundness of a certain “Grassmanian agreement tester”. In this work, we show that soundness of Grassmannian agreement tester follows from a conjecture we call the “Shortcode Expansion Hypothesis” characterizing the non-expanding sets of the degree-two Short code graph. We also show the latter conjecture is equivalent to a characterization of the non-expanding sets in the Grassman graph, as hypothesized by a follow-up paper of Dinur et al. (2017).

Following our work, Khot, Minzer and Safra (2018) proved the “Shortcode Expansion Hypothesis”. Combining their proof with our result and the reduction of Dinur et al. (2016), completes the proof of the 2 to 2 conjecture with imperfect completeness. We believe that the Shortcode graph provides a useful view of both the hypothesis and the reduction, and might be suitable for obtaining new hardness reductions.

**2012 ACM Subject Classification** Theory of computation → Problems, reductions and completeness

**Keywords and phrases** Unique Games Conjecture, Small-Set Expansion, Grassmann Graph, Shortcode

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2019.9

**Related Version** A full version of the paper is available at <https://arxiv.org/abs/1804.08662>.

**Acknowledgements** We thank Irit Dinur and Dor Minzer for comments and discussions related to this work.

## 1 Introduction

In [5], Subhash Khot put forward a family of variants of Unique Games Conjectures. A binary constraint  $P(x_1, x_2)$  where  $x_i$ s take values in alphabet  $\Sigma$  is said to be  $d$ -to- $d$  if for every value to  $x_1$ , there are exactly  $d$  values for  $x_2$  that satisfy  $P$  and vice versa. For any  $d$ ,

---

<sup>1</sup> Supported by NSF awards CCF 1565264 and CNS 1618026 and a Simons Investigator Fellowship.

<sup>2</sup> Supported by Schmidt foundation fellowship and NSF CCF-1412958.



the “ $d$ -to- $d$  games conjecture” roughly says that for every  $\epsilon > 0$ , there is some finite alphabet  $\Sigma$  such that it is NP-hard to distinguish, given a constraint satisfaction problem with  $d$ -to- $d$  constraints, whether it is possible to satisfy at least  $1 - \epsilon$  fraction of the constraints, or if every assignment satisfies at most  $\epsilon$  fraction of the constraints.<sup>3</sup> The case of  $d = 1$  corresponds to the more famous *Unique Games Conjecture*, but until recently there was no constant  $d$  for which the corresponding  $d$ -to- $d$  conjecture was known to be true.

Dinur, Khot, Kindler, Minzer, and Safra [3], building on ideas of Khot, Minzer and Safra [6], recently initiated an approach towards proving the 2-to-2 conjecture, based on a certain combinatorial hypothesis positing the soundness of the “Grassmann Agreement Test”. In this work we show that their hypothesis follows from a certain natural hypothesis characterizing the structure of non-expanding sets in the degree two Short code graph [2]. Following our work, Khot, Minzer and Safra [7] proved the latter hypothesis thus completing the proof of the 2-to-2 games conjecture. This has several implications to hardness of approximation including improving on the NP-hardness of approximation for Vertex Cover along with a host of other improved NP-hardness results. Perhaps more importantly, this also gives a strong evidence for the truth of the Unique Games Conjecture itself. We defer to [3, 4, 7] for a detailed discussion on the importance of the 2-to-2 games conjecture, as well as the reduction of this conjecture to showing the soundness of the Grassmann agreement tester.

## 1.1 Our Results

Our main result reduces the task of proving the “Grassmann Agreement Hypothesis” of Dinur, Khot, Kindler, Minzer and Safra [3, Hypothesis 3.6] to characterizing the structure of non-expanding sets in the associated Grassmann graph.

- We show that the Grassmann Agreement Hypothesis [3, Hypothesis 3.6] follows from the Grassmann Expansion Hypothesis [4, Hypothesis 1.7].
- We describe the related Shortcode test and the associated agreement and expansion hypothesis and relate them to the Grassmann versions above.

The above, combined with the work of [3, 7], suffices to prove the 2-to-2 conjecture. However we note that it is possible to directly obtain a proof of the 2-to-2 conjecture (see the recent exposition at [1]) using the “Shortcode Expansion Hypothesis” without going through the Grassmann graph at all. We think the Short code view provides a natural way to understand the reduction and suggests potential extensions, see Section 1.6.

## 1.2 Grassmann Graph and DKKMS Consistency Test

To state our results formally, we need to define the Grassman and Short code graphs, which we now do. The Grassmann graph  $\mathcal{G}(\ell, n)$  with parameters  $\ell, n$  has vertices given by all  $\ell$ -dimensional subspaces (denoted by  $\mathcal{V}_\ell$ ) of  $\mathbb{F}_2^n$ . Two subspaces  $V, V'$  of  $\mathbb{F}_2^n$  have an edge between them if  $\dim(V \cap V') = \ell - 1$ .

Let  $\text{LIN}(\mathbb{F}_2^n)$  be the set of all linear functions  $\mathbb{F}_2^n \rightarrow \mathbb{F}_2$ . For every  $f \in \text{LIN}(\mathbb{F}_2^n)$ , let  $F_f$  be the map that assigns to every  $V \in \mathcal{V}_\ell$ ,  $F_f(V) = f|_V$  the restriction of the linear function  $f$  to the subspace  $V$ . Let  $\text{LIN}(\ell, n) = \{F_f \mid f \in \text{LIN}(\mathbb{F}_2^n)\}$  be the set of all such maps.

The Grassmann Consistency test is a two-query test for  $\text{LIN}(\ell, n)$  described below:

---

<sup>3</sup> For  $d > 1$ , the conjectures are often stated in their *perfect completeness variant*, where we replace  $1 - \epsilon$  with 1 in the first case. In this work (as well as all the line of works following [6]), we refer to the imperfect completeness version as stated above.

————— **Test 1: Grassmann Consistency Test** —————

**Given:** a map  $F$  from  $\mathcal{V}_\ell \rightarrow \text{LIN}(\mathbb{F}_2^\ell)$  that maps any  $V \in \mathcal{V}_\ell$  to  $F(V)$  a linear function on  $V$ .

**Operation:**

1. Pick an edge  $(V, V')$  of  $\mathcal{G}(\ell, n)$  uniformly at random.
2. Receive  $F(V), F(V') \in \text{LIN}(\ell, n)$ .
3. Accept if  $F(V)_{V \cap V'} = F(V')_{V \cap V'}$  otherwise reject.

It is easy to see the following completeness of the Grassmann graph test.

► **Fact 1 (Completeness).** *Suppose  $F \in \text{LIN}(\ell, n)$ . Then,  $F$  passes the Grassman Consistency test with probability 1.*

The DKKMS hypothesis conjectures a precise version of soundness of the Grassmann Consistency Test.

► **Hypothesis 2 (DKKMS Soundness Hypothesis).** *For every  $\delta > 0$ , there exists  $\epsilon > 0$ , and an integer  $r > 0$  such that following holds for sufficiently large  $n \gg \ell$ .*

*Let  $F : \mathcal{V}_\ell \rightarrow \text{LIN}(\mathbb{F}_2^\ell)$  such that  $\Pr_{(V, V') \sim \mathcal{G}(\ell, n)}[F(V)_{V \cap V'} = F(V')_{V \cap V'}] \geq \delta$ . Then, there exist subspaces  $Q, W \subseteq \mathbb{F}_2^n$  of dimensions  $r$  and  $n - r$  respectively and an  $f \in \text{LIN}(\mathbb{F}_2^n)$  such that*

$$\Pr_{V \sim \mathcal{V}_\ell, Q \subseteq V \subseteq W} [F(V) = f_V] \geq \epsilon.$$

### 1.3 Shortcode Graph and Consistency Test

We now define the closely related *Degree 2 Shortcode* graph and an immediate analog of the Grassmann consistency test on this graph. For parameters  $\ell, n$  as before, the vertices of the degree 2 Short code graph  $\mathcal{S}_{\ell, n}$  are elements of  $\text{Mat}_{\ell, n}$ , that is, all matrices on  $\mathbb{F}_2$  with dimensions  $\ell \times n$ . Two vertices  $M_1$  and  $M_2$  have an edge between them if  $M_1 - M_2$  is a rank 1 matrix over the field  $\mathbb{F}_2$ . The 2 query codeword test on this graph is entirely analogous to the one above for the Grassmann graph:

————— **Test 2: Degree 2 Shortcode Consistency Test** —————

**Given:** a map  $F$  from  $\text{Mat}_{\ell, n} \rightarrow \mathbb{F}_2^\ell$ .

**Operation:**

1. Pick  $M_1 \sim \text{Mat}_{\ell, n}$  and a rank 1 matrix  $ab^\top$  for vectors  $a \in \mathbb{F}_2^\ell, b \in \mathbb{F}_2^n$  all uniformly at random from their respective domains. Let  $M_2 = M_1 + ab^\top$ .
2. Receive  $F(M_1), F(M_2) \in \mathbb{F}_2^\ell$ .
3. Accept if  $F(M_2) \in \{F(M_1), F(M_1) + a\}$ .

Just as the Grassmann consistency test, the above Short code consistency test is a “2-to-2” constraint and the following completeness is easy to establish.

► **Fact 3 (Completeness).** *Let  $f : \mathbb{F}_2^n \rightarrow \mathbb{F}_2^\ell$  be any affine linear function. Let  $F = F_f : \text{Mat}_{\ell, n} \rightarrow \mathbb{F}_2^\ell$  be the map that evaluates  $f$  on each row the input matrix. Then,  $F$  passes the Short code consistency test with probability 1.*

The analogous soundness hypothesis can now be stated as:



► **Hypothesis 4** (Degree 2 Shortcode Soundness Hypothesis). *For every  $\delta > 0$ , there exists  $\epsilon > 0$ , and an integer  $r > 0$  such that following holds for sufficiently large  $n \gg \ell$ .*

*Let  $F : \text{Mat}_{\ell,n} \rightarrow \mathbb{F}_2^\ell$  such that  $\Pr_{M \sim \text{Mat}_{\ell,n}, a \sim \mathbb{F}_2^\ell, b \sim \mathbb{F}_2^n} [F(M + ab^\top) \in \{F(M), F(M) + a\}] \geq \delta$ . Then, there exists linear constraints  $(q_i, t_i)$  and  $(r_i, s_i)$  for  $i \leq r$  and a  $z \in \mathbb{F}_2^n, u \in \mathbb{F}_2^\ell$  such that*

$$\Pr_{M \sim \text{Mat}_{\ell,n}} [F(M) = Mz + u \mid Mq_i = t_i, r_i^\top M = s_i \ \forall i \leq r] \geq \epsilon.$$

## 1.4 Soundness vs Small-Set Expansion in Grassmann/Shortcode Graphs

Recall that for a regular graph  $G$ , the expansion of a set  $S$  of vertices is the probability that a random walk beginning at a uniformly random vertex in  $S$  steps out of  $S$ . That is,  $\Phi_G(S) = \Pr_{v \sim S, v' \sim v} [v' \notin S]$ .

The DKKMS Soundness Hypothesis implies a natural characterization small non-expanding sets in  $\mathcal{G}(\ell, n)$  noted below as Hypothesis 6. Similarly, the degree 2 Short code soundness hypothesis implies a natural characterization of non-expanding sets in  $\mathcal{S}_{\ell,n}$ . We include a brief overview of the argument here and refer the reader to the more extensive commentary in Section 1.3 of [3] for further details.

Suppose  $A_1, A_2, \dots, A_r$  are “non-expanding” sets that cover a constant fraction of vertices in  $\mathcal{G}(\ell, n)$ . We construct a labeling strategy  $F$  by choosing  $r$  uniformly random linear functions  $f_i : \mathbb{F}_2^n \rightarrow \mathbb{F}_2$  and setting  $F(V) = f_i$  if  $V \sim A_i$  and  $F(V)$  is a random linear function otherwise. Clearly,  $F$  doesn’t agree with a single linear function on significantly more than  $1/r$  fraction of the vertices in  $\mathcal{V}_\ell$ . On the other hand, if  $A_i$ s are sufficiently non-expanding, then, a random edge will lie inside one of the  $A_i$ s with a non-trivially large probability and thus  $F$  will satisfy the Grassmann consistency test. In this, case, we will hope that there are subspaces  $Q, W$  of constant dimension and co-dimension, respectively such that restricting to subspaces  $V \in \mathcal{V}_\ell(Q, W)$  (where  $\mathcal{V}_\ell(Q, W)$  is the subset  $V \in \mathcal{V}_\ell$  such that  $V \subseteq W$ ) implies that  $F(V) = f_V$  for some fixed global linear function  $f$ . This can happen in the above example for  $F$  only if there are  $Q, W$  as above such that one of the  $\frac{|A_i \cap \mathcal{V}_\ell(Q, W)|}{|\mathcal{V}_\ell(Q, W)|}$  is  $\Omega(1)$  (i.e. independent of  $\ell, n$ ). Thus, Hypothesis 2 forces that the non-expanding sets  $A_i$  to be “structured” (in the sense of having a large density inside  $\mathcal{V}_\ell(Q, W)$  for some  $Q, W$  of constant dimension and co-dimension, respectively.) This can be interpreted as saying that the non-expansion of any set of vertices in  $\mathcal{G}(\ell, n)$  can be “explained” away by a more than typical density in one of the canonical non-expanding sets (i.e., those that contain a subspace  $Q$  and are contained inside a subspace  $W$  of constant dimension and co-dimension, respectively.)

To formally state the Grassmann Expansion Hypothesis, we define the special non-expanding sets (referred to as “zoom-in” and “zoom-outs” in [4]):

► **Definition 5** (Nice Sets in Grassmann Graph). A subset  $S \subseteq \mathcal{V}_\ell$  of vertices in  $\mathcal{G}(\ell, n)$  is said to be  $r$ -nice if there are subspace  $Q, W$  of  $\mathbb{F}_2^n$  of dimension and co-dimension  $r_1, r_2$  respectively such that  $r_1 + r_2 = r$  and  $S = \{V \subseteq \mathcal{V}_\ell \mid Q \subseteq V \subseteq W\}$ .

► **Hypothesis 6** (Grassmann Expansion Hypothesis). *For every  $\eta > 0$ , there exists  $\delta, r$  depending only on  $\eta$  such that if  $S \subseteq \mathcal{V}_\ell$  satisfies  $\Phi_{\mathcal{G}(\ell,n)}(S) < \eta$ , then, there are subspaces  $Q, W$  over  $\mathbb{F}_2^n$  of dimension and co-dimension  $r_1, r_2$  satisfying  $r_1 + r_2 \leq r$  respectively, such that  $\Pr_{V: Q \subseteq V \subseteq W} [V \in S] \geq \delta$ .*

Analogously, we can define nice sets in the degree 2 Short code graph and state the expansion hypothesis. We call  $Q$ , a *right* affine subspace of matrices in  $\text{Mat}_{\ell,n}$  if there are pairs  $(q_i, t_i)$  and every  $M \in Q$  satisfies  $Mq_i = t_i$ . We define a *left* affine subspace analogously.



► **Definition 7** (Nice Sets in Degree 2 Shortcode Graph). A subset  $S \subseteq \mathcal{S}_{\ell,n}$  is said to be  $r$ -nice if it is an intersection of a left and right affine subspace in  $\text{Mat}_{\ell,n}$  with sum of the dimensions  $r$ .

► **Hypothesis 8** (Shortcode Expansion Hypothesis). For every  $\eta > 0$ , there exist  $\delta, r$  depending only on  $\eta$  such that for every subset  $S \subseteq \text{Mat}_{\ell,n}$ , if  $\Pr_{M \sim S, a \sim \mathbb{F}_2^\ell, b \sim \mathbb{F}_2^n} [M + ab^\top \in S] \geq \eta$ , then, there exists an  $r$ -nice set  $\mathcal{T} \subseteq \mathcal{S}_{\ell,n}$  such that  $|S \cap \mathcal{T}| \geq \delta |\mathcal{T}|$ .

While Hypotheses 2 and 4 posit soundness of a specific “code-word consistency” test associated with the Grassmann/Shortcode graphs, Hypotheses 6 and 8 ask for a purely graph theoretic property: a characterization of non-expanding sets in  $\mathcal{G}(\ell, n)$  and  $\mathcal{S}_{\ell,n}$ . As such, it appears easier to attack and [3] thus suggested understanding the structure of non-expanding sets in  $\mathcal{G}(\ell, n)$  as a natural first step. As we show in this note, proving Hypothesis 8 is in fact enough to show Hypothesis 2. In a follow up work [7], this result was used in to complete the proof of the DKKMS soundness hypothesis.

## 1.5 Our Results

We are now ready to state our main results formally.

First, we show that the soundness of the Short code consistency test follows from the expansion hypothesis for the Short code graph.

► **Theorem 9.** *The degree 2 Shortcode Expansion Hypothesis 8 implies the Degree 2 Shortcode Soundness Hypothesis 4.*

Second, we show that the soundness hypothesis for the Short code consistency test implies the soundness hypothesis for the Grassmann consistency test. This reduces the DKKMS soundness hypothesis to establishing the expansion hypothesis for the Shortcode graph.

► **Theorem 10.** *The degree 2 Shortcode Soundness Hypothesis implies the Grassmann Soundness Hypothesis 2.*

Finally, we relate the expansion hypothesis of the Grassmann graph to the expansion hypothesis for the degree 2 Short code graph.

► **Theorem 11.** *The Grassmann Expansion Hypothesis (Hypothesis 6) is equivalent to the Shortcode Expansion Hypothesis (Hypothesis 8).*

## 1.6 Discussion

Working with the Short code consistency test (and consequently, the Short code expansion hypothesis) makes an approach to proving Hypothesis 2 somewhat more tractable. This is because unlike the Grassmann graph, Degree 2 Short code graph is a Cayley graph on  $\text{Mat}_{\ell,n}$  under the group operation of  $\mathbb{F}_2$ -addition with the set of all rank 1 matrices forming the set of generators. Thus studying expansion of sets of vertices can be approached via powerful methods from Fourier analysis. Indeed, this is the route taken by the recent breakthrough [7] that proves the Short code expansion hypothesis and completes the proof of the 2-to-2 games conjecture (with imperfect completeness).

Perhaps equally importantly, the Short code consistency test suggests immediate extensions (*higher degree Short code graphs*) that provide a natural path to proving the Unique Games Conjecture. We discuss this approach here.

First, the Grassmann/Short code consistency tests as stated above are “2-to-2” tests. That is, for any reply for the first query, there are two admissible replies for the other query.

However, it is simple to modify the tests and make them *unique* or “1-to-1” at the cost of making the completeness  $1/2$  instead of  $1$ . For concreteness, we describe this simple modification below.

**Test 3: Unique Degree 2 Shortcode Consistency Test**

**Given:** a map  $F$  from  $\text{Mat}_{\ell,n} \rightarrow \mathbb{F}_2^\ell$ .

**Operation:**

1. Pick  $M_1 \sim \text{Mat}_{\ell,n}$  and a rank 1 matrix  $ab^\top$  for vectors  $a \in \mathbb{F}_2^\ell$ ,  $b \in \mathbb{F}_2^n$  all uniformly at random from their respective domains. Let  $M_2 = M_1 + ab^\top$ .
2. Receive  $F(M_1), F(M_2) \in \mathbb{F}_2^\ell$ .
3. Accept if  $F(M_2) = F(M_1)$ .

**Test 4: Unique Degree 3 Shortcode Consistency Test**

**Given:** a map  $F$  from  $\text{Ten}_{\ell,m,n} \rightarrow \mathbb{F}_2^\ell$ .

**Operation:**

1. Pick  $T_1 \sim \text{Ten}_{\ell,m,n}$  and a rank 1 tensor  $a \otimes b \otimes c$  for vectors  $a \in \mathbb{F}_2^\ell$ ,  $b \in \mathbb{F}_2^m$  and  $c \in \mathbb{F}_2^n$  all uniformly at random from their respective domains. Let  $T_2 = T_1 + a \otimes b \otimes c$ .
2. Receive  $F(T_1), F(T_2) \in \mathbb{F}_2^\ell$ .
3. Accept if  $F(T_2) = F(T_1)$ .

It is easy to check that the any strategy that passes the 2-to-2 test can be modified to obtain a success probability of  $1/2$  in passing the “unique” test above (see proof of Lemma 14 below). This is one of the several ways that the NP hardness of “2-to-2” games implies the NP hardness of  $(1/2, \epsilon)$ -unique games - that is, distinguishing between instances where at least  $1/2$  the constraints are satisfiable from those where at most  $\epsilon$  fraction of constraints are satisfiable.

A natural strategy, thus, to try to show NP hardness of  $(1 - \epsilon, \epsilon)$ -unique games is to use some variant of the Short code consistency test above that has completeness  $1 - \epsilon$  instead of  $1/2$ . Indeed, the degree 2 Short code consistency test suggests natural analogs with higher completeness - by moving to higher degree Short code graphs. For concreteness, consider the following test on degree 3 Short code graphs, where it is easy to argue a completeness of  $3/4$ .

Let  $\text{Ten}_{\ell,m,n}$  be the set of all  $\ell \times m \times n$  tensors over  $\mathbb{F}_2$ . Recall that a rank 1 tensor is defined by 3 vectors  $a \in \mathbb{F}_2^\ell$ ,  $b \in \mathbb{F}_2^m$  and  $c \in \mathbb{F}_2^n$  and can be written as  $a \otimes b \otimes c$ .

To see why there’s a natural analog of the strategy in case of the degree 2 Short code consistency test that gives a completeness of  $3/4$ , we show:

► **Lemma 12 (Completeness).** *Let  $y \in \mathbb{F}_2^m$  and  $z \in \mathbb{F}_2^n$ . Let  $F_f : \text{Ten}_{\ell,m,n} \rightarrow \mathbb{F}_2^\ell$  be the map that assigns to any tensor  $T$ , the value  $F_f(T)_i = \sum_{j,k} T(i, j, k)y_j z_k$ . Then,  $F_f$  passes the test with probability  $3/4$ .*

**Proof.** Let  $T, T'$  be such that  $T - T'$  is rank 1 tensor. Then,  $F_f$  passes the test only if  $F_f(T - T') = 0$ . If  $T - T' = a \otimes b \otimes c$ , then  $F_f(T - T') = \langle b, y \rangle \cdot \langle c, z \rangle a$ . Since  $b, c$  are independently chosen in the test, the probability that  $F_f(T - T') = 0$  is  $3/4$ . ◀

Thus, the degree 3 Short code consistency test gives a natural analog of the degree 2 Short code consistency test with higher completeness. Indeed, degree  $r$  version gives a test with completeness of  $1 - 2^{-r}$  as expected. One can also frame expansion hypotheses similar to the ones for the degree 2 case that posit a characterization of the non-expanding sets in higher degree Short code graphs.

While our current efforts to compose this test with the “outer-PCP” in order to get a reduction to Unique Games problem (with higher completeness) have not succeeded, it seems a natural avenue for launching an attack on the UGC.<sup>4</sup>

## 2 Small-Set-Expansion vs Soundness

In this section, we establish that the inverse Short code hypothesis (Hypothesis 8) implies the soundness of the degree 2 Short code consistency test 4.

### From 2-to-2 to Unique Tests

For the sake of exposition, it will be easier to work with Test 1.6, the “unique” version of the degree 2 Short code consistency test. Thus, we restate the soundness hypothesis for Test 1.6 and show that it is enough to establish Hypothesis 4.

► **Hypothesis 13** (Soundness of Test 1.6). *For every  $\eta > 0$ , there exists  $\delta > 0$ , and an integer  $r > 0$  such that following holds for sufficiently large  $n \gg \ell$ .*

*Let  $F : \text{Mat}_{\ell,n} \rightarrow \mathbb{F}_2^\ell$  such that  $\Pr_{M \sim \text{Mat}_{\ell,n}, a \sim \mathbb{F}_2^\ell, b \sim \mathbb{F}_2^n} [F(M + ab^\top) = F(M)] \geq \eta$ . Then, there exists linear constraints  $(q_i, t_i)$  and  $(r_i, s_i)$  for  $i \leq r$  and a  $z \in \mathbb{F}_2^n, u \in \mathbb{F}_2^\ell$  such that*

$$\Pr_{M \sim \text{Mat}_{\ell,n}} [F(M) = Mz + u \mid Mq_i = t_i, r_i^\top M = s_i \forall i \leq r] \geq \delta.$$

We first show that Hypothesis 13 implies Hypothesis 4.

► **Lemma 14.** *Hypothesis 13 implies Hypothesis 4.*

**Proof.** Let  $F$  be the labeling strategy for Test 1.3. We will first obtain a good labeling strategy for Test 1.6 by modifying  $F$  slightly.

Choose  $h$  uniformly at random from  $\mathbb{F}_2^n$ . For any  $M \in \text{Mat}_{\ell,n}$ , let  $G(M) = F(M) + Mh$ . We claim that if  $F$  passes the Test 1.3 with probability  $\eta$ , then  $G$  passes Test 1.6 with probability at least  $\eta/2$ .

To see this, take any  $M, M'$  such that  $M \sim M'$  in  $\mathcal{S}_{\ell,n}$ . That is,  $M - M' = ab^\top$  for vectors  $a, b$ . We will argue that  $G(M) = G(M')$  with probability  $1/2$ . This will imply that in expectation over the choice of  $h$ ,  $G$  satisfies at least  $1/2$  the constraints satisfied by  $F$  in Test 1.3 completing the proof.

This is simple to see: since  $F$  passes the test,  $F(M) = F(M')$  or  $F(M) - F(M') = a$ . WLOG, say the first happens. Observe that  $G$  passes the unique test on  $M, M'$  if  $F(M) + Mh = F(M') + M'h$  or  $F(M) - F(M') = (M - M')h = \langle b, h \rangle a$ . Since  $F(M) = F(M')$ ,  $G$  thus passes if  $\langle b, h \rangle = 0$  which happens with probability  $1/2$ . ◀

<sup>4</sup> There are indeed very serious obstacles that must be overcome before carrying this out. Specifically, the reduction of [3] uses a careful interplay between *smoothness* properties of the outer PCP and *efficiency* or “blow up” properties of the test (i.e., the number of potential queries by the verifier as a function of the number of honest strategies). The tensor based test has too much of a blowup to be able to be simply “plugged in” in the outer PCP used by [3].

### Expansion to Soundness

We will now show that Hypothesis 8 implies Hypothesis 13. This completes the proof of Theorem 9. A similar argument can be used to directly establish that Hypothesis 6 implies Hypothesis 2. We do not include it here explicitly. Instead, we relate the expansion and soundness hypothesis for the degree 2 Short code test to the analogs for the Grassmann test as we believe this could shed light on showing expansion hypotheses for higher degree Short code tests discussed in the next section.

► **Lemma 15.** *Hypothesis 8 implies Hypothesis 13.*

**Proof.** Let  $F$  be the labeling function as in the assumption in Hypothesis 13. Then, we know that  $\Pr_{M \sim \text{Mat}_{\ell,n}, a \sim \mathbb{F}_2^\ell, b \sim \mathbb{F}_2^n} [F(M) = F(M + ab^\top)] \geq \eta$ . For any  $z \in \{0, 1\}^\ell$ , let  $S_z$  be the set of all matrices  $M$  with  $F(M) = z$ . Then, by an averaging argument, there must be a  $z \in \{0, 1\}^\ell$  such that  $\Pr_{M \sim S_z, a \sim \mathbb{F}_2^\ell, b \sim \mathbb{F}_2^n} [M + ab^\top \in S_z] \geq \eta$ .

Apply Hypothesis 8 to  $S_z$  to obtain  $r$ -nice subset  $Q$  of  $\text{Mat}_{\ell,n}$  such that  $|Q \cap S_z| \geq \delta|Q|$ . Let  $Mq = t$  be an affine constraint satisfied by every  $M \in Q$ . Consider the affine linear strategy  $H : \text{Mat}_{\ell,n} \rightarrow \mathbb{F}_2^\ell$  that maps any  $M$  to  $H(M) = Mq + t + z$ . Observe that for every  $M \in Q$ ,  $H(M) = z$  by this choice. As a result, when  $M \sim M'$  are such that  $M, M' \subseteq Q$ ,  $\Pr[H(M) = H(M')] \geq \delta$ . Thus,  $H$  is the “decoded” strategy that satisfies the requirements of Hypothesis 13 as required. This completes the proof. ◀

## 3 Relating Grassmann Graphs to Degree 2 Shortcode Graphs

In this section, we show a formal relationship between the Grassmann and the degree 2 Shortcode tests. In particular, we will prove Theorems 10 and 11.

### 3.1 A homomorphism from $\mathcal{G}(\ell, n)$ into $\mathcal{S}_{\ell,n}$

Key to the relationship between the two tests is an embedding of the degree 2 Short code graph  $\mathcal{S}_{\ell,n}$  into  $\text{Mat}_{\ell,n-\ell}$ . We describe this embedding first. As justified in the previous section, it is without loss of generality to work with the “unique” versions of both the tests.

To describe the above embedding, we need the notion of *projection* of a subspace of  $\mathbb{F}_2^n$  to a set of coordinates.

► **Definition 16** (Projection of a Subspace). Given a subspace  $V \subseteq \mathbb{F}_2^n$ , the projection of  $V$  to a set of coordinates  $S \subseteq [n]$ , written as  $\text{Proj}_S(V)$  is the subspace of  $\mathbb{F}_2^{|S|}$  defined by taking the vectors obtained by keeping only the coordinates indexed by  $S$  for every vector  $v \in V$ .

Let  $\mathcal{B} \subseteq \mathbb{F}_2^n$  be the set  $n$ -tuples of linearly independent elements of  $\mathbb{F}_2^n$ , i.e. each  $B \in \mathcal{B}$  forms a basis for the vector space  $\mathbb{F}_2^n$ . We will use  $B_0$  to denote the standard basis  $\{e_1, e_2, \dots, e_n\}$ .

We will now describe a class of graph homomorphisms from  $\mathcal{G}(\ell, n)$  into  $\mathcal{S}_{\ell,n-\ell}$ . Each element of this class can be described by a basis  $B$  of  $\mathbb{F}_2^n$ .

For each basis  $B \in \mathcal{B}$ , let  $\mathcal{V}_\ell(B) \subseteq \mathcal{V}_\ell$  be the set of all subspaces  $V \in \mathcal{V}_\ell$  such that the projection of  $V$  to the first  $\ell$  coordinates when written w.r.t. the basis  $B$  is full-dimensional. Our embedding will map each element of  $\mathcal{V}_\ell(B)$  into a distinct element of  $\text{Mat}_{\ell,n}$  such that the edge structure within  $\mathcal{V}_\ell(B)$  in  $\mathcal{G}(\ell, n)$  is preserved under this embedding.

► **Definition 17** (Homomorphism from  $\mathcal{G}(\ell, n)$  into  $\mathcal{S}_{\ell,n}$ ). Let  $\phi = \phi_B : \mathcal{V}_\ell(B) \rightarrow \text{Mat}_{\ell,n-\ell}$  be defined as follows. Write every vector in the  $B$ -basis. For any  $V \in \mathcal{V}_\ell(B)$  and for  $1 \leq i \leq \ell$ , let  $v_i$  be the unique vector in  $V$  such that  $\text{Proj}_{[i]}(v_i) = e_i \in \mathbb{F}_2^\ell$ . We call  $v_1, v_2, \dots, v_\ell$  to be the *canonical basis* for  $V$ .

Define  $\phi(V)$  to be the  $\ell \times (n - \ell)$  matrix with the  $i^{\text{th}}$  row given by the projection of  $v_i$  on the last  $(n - \ell)$  coordinates for each  $1 \leq i \leq \ell$ . When the basis  $B$  is clear from the context, we will omit the subscript and write  $\phi$ .

It is easy to confirm that  $\phi$  is a bijection from  $\mathcal{V}_\ell(B)$  into  $\text{Mat}_{\ell, n}$ . This is because canonical basis for a subspace  $V$  is unique.

Next, we prove some important properties of the homomorphism  $\phi$  that will be useful in the proof of Theorem 10.

First, we show that the map  $\phi$  is indeed a homomorphism as promised and thus, preserves edge structure.

► **Lemma 18** ( $\phi$  is a homomorphism). *For  $\phi = \phi_B$  defined above and any  $V, V' \in \mathcal{V}_\ell(B)$ ,  $V \sim V'$  in  $\mathcal{G}(\ell, n)$  iff  $\phi(V) \sim \phi(V')$  in  $\mathcal{S}_{\ell, n}$ .*

**Proof.** Let  $u \in GF(2)^\ell, v \in GF(2)^{n-\ell}$  be arbitrary non-zero vectors that define a rank 1 matrix  $uv^\top$ . Consider the matrix  $M = M_V + uv^\top$ . Then,  $M \in \text{Mat}_{\ell, n-\ell}$  and thus  $\phi^{-1}(M) = W \in \mathcal{V}_\ell(B)$ . We claim that  $\dim(W \cap V) = \ell - 1$ . Suppose  $b_1, b_2, \dots, b_\ell$  are the rows of  $M_V$ . Then, the rows of  $M$  are given by  $b_i + u_i v$ . Thus,  $W$  is spanned by  $(e_i, b_i + u_i v)$  where  $e_i$  is the  $i^{\text{th}}$  standard basis element on the first  $\ell$  coordinates and the notation  $(e_i, b_i + u_i v)$  indicates the concatenation of the vectors in the ordered pair to get a  $n$  dimensional vector. In particular, every element of  $W$  can be written as  $\sum_{i \leq \ell} \lambda_i (e_i, b_i) + (\sum_{i \leq \ell} \lambda_i u_i) v$  and any such vector is contained in  $V$  if  $(\sum_{i \leq \ell} \lambda_i u_i)$  implying that  $\dim(V \cap W) = \dim(V) - 1 = \ell - 1$ .

On the other hand, let  $V'$  be a subspace in  $\mathcal{V}_\ell(B)$  such that  $V' \sim V$  and let  $M_V$  and  $M_{V'}$  be the matrices obtained via the map  $\phi$ . Then,  $M_V$  and  $M_{V'}$  must differ in at least one row, say, WLOG, the last row of  $M_V$  and  $M_{V'}$  are  $(e_\ell, v)$  and  $(e_\ell, v')$  respectively. Notice that since the vector with  $e_\ell$  in the first  $\ell$  coordinates is unique in  $V, V'$ , neither of  $(e_\ell, v), (e_\ell, v')$  belong to the intersection  $V \cap V'$ . Further, for every vector  $z \in V$ , either  $z$  or  $z + (e_\ell, v)$  must be contained in the intersection  $V \cap V'$  (as the extra linear equation that  $V \cap V'$  satisfies over and above  $V$  is satisfied by exactly one of  $z$  and  $z + (e_\ell, v)$ ). Thus, by letting  $b'_i = b_i + (e_\ell, v) + (e_\ell, v')$  to every one of the canonical basis elements  $b_i$  of  $V$  that are not in  $V \cap V'$ , we get a set of elements that are all 1) contained in  $V'$  2)  $\text{Proj}_{[\ell]} b'_i = e_i$  for every  $i$ . This then has to be the canonical basis of  $M_{V'}$  (by uniqueness of the canonical basis) and further, the corresponding  $M_{V'}$  can be written as  $1_S(w + w')^\top$  where  $S$  is the set of  $i$  such that  $b_i$  is not in  $V \cap V'$ . ◀

Next, we want to argue that expansion of sets is preserved up to constant factors under the map  $\phi$ . Towards this, we first show that  $\mathcal{V}_\ell(B)$  contains a fraction of the vertices of  $\mathcal{G}(\ell, n)$  as we next show.

► **Lemma 19** (Projections of Subspaces). *Let  $V \sim \mathcal{V}_\ell$  for  $\ell \leq \sqrt{n}/2$ . Then,  $\Pr[\dim \text{Proj}_{[\ell]}(V) = \ell] \geq 0.288$  for large enough  $n$  and  $\ell = \omega(1)$ .*

*Further, let  $V \in \mathcal{V}_\ell(B)$  for some  $B$ . Then, at least 1/2 fraction of the neighbors of  $V$  in  $\mathcal{G}(\ell, n)$  are contained in  $\mathcal{V}_\ell(B)$ .*

**Proof.** We can sample a random subspace of  $\ell$  dimension as follows: Choose  $\ell$  uniformly random and independent points from  $GF(2)^n$ . If they are linearly independent, let  $V$  be the subspace spanned by them.

We can estimate the probability that the sampled points are linearly independent as:  $\prod_{i=0}^{\ell-1} (1 - 2^{-n+i}) \geq 1 - 2^{-n} 2^{\ell^2}$ .

Next, we estimate the probability that the projection to first  $\ell$  coordinates of the sampled vectors is linearly independent. By a similar reasoning as above, this probability is at least  $\prod_{i=0}^{\ell-1} (1 - 2^{-\ell+i}) \approx 0.289$  (the limit of this product for large  $\ell$ .)

By a union bound, thus, a random subspace has a full dimensional projection on  $S$  with probability at least  $0.289 - 2^{-n/2}$  for any  $\ell < \sqrt{n}/2$ .

For the remaining part, assume that  $B = B_0$  - the standard basis. Notice that a random neighbor of  $V$  can be sampled as follows: choose a uniformly random basis for  $V$ , say  $v_1, v_2, \dots, v_\ell$ . Replace  $v_\ell$  by a uniformly random vector  $v'_\ell$  outside of  $V$  in  $\mathbb{F}_2^n$ . Since  $V \in \mathcal{V}_\ell(B)$ , the projection of  $V$  to the first  $\ell$  coordinates is linearly independent.  $V'$  would thus satisfy the same property whenever  $v_\ell$  is such that the projection of  $v'_\ell$  to the first  $\ell$  coordinates is not in the span of the projection to the first  $\ell$  coordinates of  $v_1, v_2, \dots, v_{\ell-1}$ . The chance of this happening is exactly  $1/2$ . This completes the proof.  $\blacktriangleleft$

As a consequence of above, we can now obtain that the preimages of non-expanding sets under  $\phi$  are non-expanding in  $\mathcal{G}(\ell, n)$ .

► **Lemma 20.** *Let  $T \subseteq \text{Mat}_{\ell, n}$  be a subset satisfying  $\Pr_{M \sim T, M' \sim M}[M' \in T] = \eta$ . Then,  $\phi^{-1}(T)$  satisfies:  $\Pr_{V \sim \phi^{-1}(T), V' \sim V}[V' \in \phi^{-1}(T)] \geq \eta/2$ .*

**Proof.** Let  $B$  the basis used to construct  $\phi$ . Then,  $\phi(T) \subseteq \mathcal{V}_\ell(B)$ . By Lemma 19,  $1/2$  the neighbors of  $\phi(T)$  are contained in  $\mathcal{V}_\ell(B)$ . By assumption,  $\eta$  fraction of these neighbors are contained inside  $T$ . This finishes the proof.  $\blacktriangleleft$

Via a similar application of Lemma 19, we can establish an appropriate converse.

► **Lemma 21.** *Let  $S \subseteq \mathcal{V}_\ell$  be a subset satisfying  $\Pr_{V \sim S, V' \sim V}[V' \in S] \geq \eta$ . Then, for a uniformly random choice of basis  $B$  for  $\mathbb{F}_2^n$ ,  $\mathbb{E}_B |\phi(S \cap \mathcal{V}_\ell(B))| = \Omega(|S|)$  and  $\Pr_{M, M' \sim \phi(S \cap \mathcal{V}_\ell(B)), M' \sim M}[M' \in \phi(S \cap \mathcal{V}_\ell(B))] \geq \Omega(\eta)$ .*

Finally, we show that  $r$ -nice sets in  $\mathcal{G}(\ell, n)$  get mapped to  $r$ -nice sets in  $\mathcal{S}(\ell, n)$  and vice versa.

► **Lemma 22.** *Let  $S \subseteq \mathcal{V}_\ell$  be an  $r$ -nice set in  $\mathcal{G}(\ell, n)$ . Then,  $\phi_B(S \cap \mathcal{V}_\ell(B))$  is an  $r$ -nice set in  $\mathcal{S}_{\ell, n}$ . Conversely, if  $T \subseteq \text{Mat}_{\ell, n}$  is an  $r$ -nice set in  $\mathcal{S}_{\ell, n}$  then  $\phi^{-1}(T) = Q \cap \mathcal{V}_\ell(B)$  for some  $r$ -nice set  $Q$  in  $\mathcal{G}(\ell, n)$ .*

**Proof.** WLOG, assume that  $B = B_0$ . We will assume that  $S \subseteq \mathcal{V}_\ell$  is the set of all subspaces in  $\mathcal{V}_\ell$  contained in a subspace  $W$  of co-dimension  $r$ . The general case is analogous. Equivalently, if  $w_1, w_2, \dots, w_r$  form a basis for  $W$ , then, for every  $V \in S \cap \mathcal{V}_\ell(B)$  and ever  $v \in V$   $\langle v, w_i \rangle = 0$  for every  $i$ .

Consider the canonical basis  $v_1, v_2, \dots, v_\ell$  for  $V$  - recall that this means that the projection of  $v_i$  to the first  $\ell$  coordinates equal  $e_i$ . Thus, for every  $i$ , we can write  $v_i = (e_i, v'_i)$  for some vectors  $v'_i$  of  $n - \ell$  dimensions.

Then,  $\phi(V)$  is the matrix  $M_V$  with rows  $v'_i$  by our construction. In particular, this means that the  $M_V$  satisfies the constrain:  $M_V \cdot w_i = t_i$  where  $t_i$  is the vector with  $j$ th coordinate equal to  $\langle e_j, w_i \rangle$ . Thus, we have shown that for every  $V \in S$ ,  $\phi(V)$  satisfies a set of  $r$  affine linear equations.

Conversely, observe that if any  $M$  satisfies the affine linear equation  $M_V w_i = t_i$  as above, the set of all  $(e_i, u_i)$  for  $i \leq \ell$  where  $u_i$  is the  $i$ th row of  $M_V$ , must span a subspace in  $S$ . This yields that  $\phi(S \cap \mathcal{V}_\ell(B))$  is an  $r$ -nice set.

The converse follows from entirely similar ideas. Suppose  $T \subseteq \text{Mat}_{\ell, n}$  is an  $r$ -nice set. WLOG, we restrict to the case where  $T$  is the set of all matrices satisfying linear constraints  $M q_i = t_i$  for some choice of  $r$  linearly independent constraints  $(q_i, t_i)$ . Letting  $u_1, u_2, \dots, u_\ell$  be the rows of  $M$ , this implies that every vector  $v$  in the span of  $(e_i, u_i)$  for  $i \leq \ell$  satisfies

the linear equation  $\langle q, v \rangle = 0$  where  $q = (q_i, t_i(1), t_i(2), \dots, t_i(\ell))$ . This immediately yields that  $\phi^{-1}(M)$  is contained in a subspace  $W$  of co-dimension  $r$ . Conversely, it is easy to check that for every subspace  $V$  of dimension  $\ell$  contained in  $W \cap \mathcal{V}_\ell(B)$ ,  $\phi(V)$  satisfies the  $r$  affine linear constraints above.

This completes the proof. ◀

### 3.2 Shortcode Test vs Grassmann Test

We now employ the homomorphism constructed in the previous subsection to relate the soundness and expansion hypothesis in Short code and Grassmann tests.

First, we show that the soundness hypothesis for degree 2 Short code consistency test implies the soundness hypothesis for the Grassmann consistency test and complete the proof of Theorem 10.

► **Lemma 23.** *The degree 2 Short code soundness hypothesis (Hypothesis 13) implies the Grassmann soundness hypothesis (Hypothesis 2).*

**Proof.** Let  $F$  be the assumed labeling strategy in Hypothesis 2. We will construct a labeling strategy for  $\mathcal{S}_{\ell,n}$  from  $G$  so that we can apply the conclusion of 13. We will first choose an embedding of the type we constructed before in order to construct  $G$ .

Let  $B \sim \mathcal{B}$  be chosen uniformly at random and let  $\phi = \phi_B$  as in the previous subsection. For any  $V \in \mathcal{V}_\ell(B)$ , let  $F(V) = f$ , a linear function restricted to  $V$ . Let  $v_1, v_2, \dots, v_\ell$  be the canonical basis for  $V$ , i.e., the projection of  $v_i$  to the first  $\ell$  coordinates (when written in basis  $B$ ) equals  $e_i$  for every  $i$ . Set  $G(\phi(V)) = z$  where  $z_i = f(v_i)$ . Since  $\phi$  is onto, this defines a labeling strategy for all of  $\text{Mat}_{\ell,n}$ .

Next, we claim that if  $F$  passes the Grassmann consistency test with probability  $\eta$  then  $G$  passes the degree 2 Short code consistency test with probability  $\Omega(\eta)$ .

Before going on to the proof of this claim, observe that this completes the proof of the lemma. To see this, we first apply Hypothesis 13 to conclude that there's an  $r$ -nice set  $Q$  in  $\mathcal{S}_{\ell,n}$  and an affine function defined by  $z \in \mathbb{F}_2^{n-\ell}, u \in \mathbb{F}_2^\ell$  such that the labeling strategy  $H(M) = Mz + u$  passes the degree 2 Short code consistency test with probability  $\delta$  for all  $M$  in  $Q$ . It is easy to construct the analogous linear strategy for the Grassmann consistency test: For any  $V \in \mathcal{V}_\ell(B)$  with the canonical basis  $v_1, v_2, \dots, v_\ell$  defined above, set  $f(v_i) = u_i + \langle v_i, z \rangle$ . Extend  $f$  linearly to the span of all such vectors. Finally, extend  $f$  to all vectors by taking any linear extension. From Lemma 19,  $1/2$  the neighbors of vertices in  $\phi^{-1}(Q)$  are contained in  $\mathcal{V}_\ell(B)$ . From Lemma 22,  $\phi^{-1}(Q) = \mathcal{F} \cap \mathcal{V}_\ell(\mathcal{B})$  for some  $r$ -nice set  $\mathcal{F}$  in  $\mathcal{G}(\ell, n)$ . Finally, by an argument similar to the one in Lemma 19,  $|\mathcal{F} \cap \mathcal{V}_\ell(\mathcal{B})| \geq \otimes(|\mathcal{F}|)$  with high probability over the draw of  $B$ . Combining the above three observations yields that  $f$  passes the Grassmann consistency test when restricted to the nice set  $\mathcal{F}$  with probability  $\Omega(\delta)$ .

We now complete the proof of the claim. This follows immediately if we show that for any  $V \sim V'$  chosen from  $\mathcal{V}_\ell(B)$ ,  $\Pr_{V \sim V', V, V' \in \mathcal{V}_\ell(B)}[F(V)|_V = F(V')|_{V'}] \geq 0.07(\eta - 2^{-n+\ell})$ .

Without loss of generality, we assume that  $B$  is the standard basis  $\{e_1, e_2, \dots, e_n\}$ . First, notice that  $\text{Span}\{V \cup V'\}$  is of dimension  $\ell + 1$  for all but  $2^{-n+\ell}$  fraction of pairs  $V \sim V'$ . Thus, we can assume that  $\Pr_{V \sim V' | \dim \text{Span}\{V \cup V'\} = \ell + 1}[F(V)|_V = F(V')|_{V'}] \geq \eta - 2^{-n+\ell}$ .

Let  $C = B^{-1}$ , the basis change matrix corresponding to  $B$  and let  $C_i$  be the  $i^{\text{th}}$  row of  $C$  and let  $C_{[\ell]}$  be the matrix formed by taking the first  $\ell$  rows of  $C$ . Fix  $V \sim V'$  for some  $V, V' \in \mathcal{V}_\ell$ . Assume now that  $\text{Span}\{V \cup V'\}$  is of dimension  $\ell + 1$ . Let  $v_1, v_2, \dots, v_{\ell-1}$  be a basis for  $V \cap V'$ . Let  $V = \{V \cap V' \cup w_1\}$  and  $V' = \{V \cap V' \cup w_2\}$  for some  $w_1, w_2$  that linearly independent of each other and of any vector in  $V \cap V'$ . We estimate the probability that



$V, V' \in \mathcal{V}_\ell(B)$ . Then, this is the probability that  $v_1, v_2, \dots, v_{\ell-1}, w_1, w_2$  are mapped by  $C^{[\ell]}$  into  $a_1, a_2, \dots, a_{\ell-1}, a_\ell, a_{\ell+1}$  respectively, satisfying  $a_\ell, a_{\ell+1} \notin \text{Span}\{a_i \mid i \leq \ell-1\}$ . It is easy to check that the probability of this over the random choice of  $B$  is at least  $0.288 * 1/4 > 0.07$ . This proves the claim.

By taking  $n$  large enough (compared to  $\ell$ ), this probability can be made larger than, say,  $0.06\eta$  (say). This finishes the proof.  $\blacktriangleleft$

Next, we show that the Grassmann Expansion Hypothesis (Hypothesis 6) is equivalent to the Shortcode Expansion Hypothesis (Hypothesis 8) and complete the proof of Theorem 11.

► **Lemma 24.** *The Grassmann Expansion Hypothesis (Hypothesis 6) is equivalent to the Shortcode Expansion Hypothesis (Hypothesis 8).*

**Proof.** First, we show that Hypothesis 6 implies Hypothesis 8.

Let  $S \subseteq \text{Mat}_{\ell,n}$  be such that  $\Pr_{M \sim S, a \in \mathbb{F}_2^\ell, b \in \mathbb{F}_2^n} [M + ab^\top \in S] = \eta$ . Then, by Lemma 20,  $\phi_B^{-1}(S)$  has an expansion of  $\Omega(\eta)$  in  $\mathcal{G}(\ell, n)$ .

Applying the Grassmann expansion hypothesis (Hypothesis 6), we know that there exists a  $r$ -nice set  $\mathcal{F}$  in  $\mathcal{G}(\ell, n)$  such that  $|\mathcal{F} \cap \phi_B^{-\infty}(S)| \geq \delta|\mathcal{F}|$ . Further, since  $\phi_B^{-1}(S) \subseteq \mathcal{V}_\ell(B)$ , we must have:  $|(\mathcal{F} \cap \mathcal{V}_\ell(B)) \cap \phi_B^{-\infty}(S)| \geq \delta|\mathcal{F} \cap \phi_B^{-\infty}(S)|$ . To finish, observe that by Lemma 22,  $\phi(\mathcal{F} \cap \phi_B^{-\infty}(S))$  is an  $r$ -nice set, say  $\mathcal{Q}$  in  $\mathcal{S}_{\ell,n}$ . This, show that  $|S \cap \mathcal{Q}| \geq \delta|\mathcal{Q}|$  completing the proof.

The proof of the other direction, that is, Hypothesis 8 implies Hypothesis 6, is analogous and relies on the use of Lemma 21.  $\blacktriangleleft$

---

## References

- 1 Mitali Bafna, Chi-Ning Chou, and Zhao Song. An Exposition of Dinur-Khot-Kindler-Minzer-Safra Proof for the 2-to-2 Games Conjecture, 2018. URL: <http://boazbarak.org/dkkmsnotes.pdf>.
- 2 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.
- 3 Irit Dinur, Subhash Khot, Guy Kindler, Dor Minzer, and Muli Safra. Towards a Proof of the 2-to-1 Games Conjecture? *Electronic Colloquium on Computational Complexity (ECCC)*, 23:198, 2016. URL: <http://eccc.hpi-web.de/report/2016/198>.
- 4 Irit Dinur, Subhash Khot, Guy Kindler, Dor Minzer, and Muli Safra. On Non-Optimally Expanding Sets in Grassmann Graphs. *STOC*, 24:94, 2018.
- 5 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. doi:10.1109/CCC.2002.1004334.
- 6 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. doi:10.1145/3055399.3055432.
- 7 Subhash Khot, Dor Minzer, and Muli Safra. Pseudorandom Sets in Grassmann Graph have Near-Perfect Expansion. *Electronic Colloquium on Computational Complexity (ECCC)*, 25:6, 2018. URL: <https://eccc.weizmann.ac.il/report/2018/006>.



# Algorithms, Bounds, and Strategies for Entangled XOR Games

Adam Bene Watts<sup>1</sup>

MIT Center for Theoretical Physics, 77 Massachusetts Ave, 6-304, Cambridge, MA, USA  
abenewat@mit.edu

Aram W. Harrow<sup>2</sup>

MIT Center for Theoretical Physics, 77 Massachusetts Ave, 6-304, Cambridge, MA, USA  
aram@mit.edu

Gurtej Kanwar<sup>3</sup>

MIT Center for Theoretical Physics, 77 Massachusetts Ave, 6-304, Cambridge, MA, USA  
gurtej@mit.edu

Anand Natarajan<sup>4</sup>

California Institute of Technology, 1200 E. California Blvd, Pasadena, CA, USA  
anandn@caltech.edu

---

## Abstract

Entangled games are a quantum analog of constraint satisfaction problems and have had important applications to quantum complexity theory, quantum cryptography, and the foundations of quantum mechanics. Given a game, the basic computational problem is to compute its *entangled value*: the supremum success probability attainable by a quantum strategy. We study the complexity of computing the (commuting-operator) entangled value  $\omega^*$  of entangled XOR games with any number of players. Based on a duality theory for systems of operator equations, we introduce necessary and sufficient criteria for an XOR game to have  $\omega^* = 1$ , and use these criteria to derive the following results:

1. An algorithm for symmetric games that decides in polynomial time whether  $\omega^* = 1$  or  $\omega^* < 1$ , a task that was not previously known to be decidable, together with a simple tensor-product strategy that achieves value 1 in the former case. The only previous candidate algorithm for this problem was the Navascués-Pironio-Acín (also known as noncommutative Sum of Squares or ncSoS) hierarchy, but no convergence bounds were known.
2. A family of games with three players and with  $\omega^* < 1$ , where it takes doubly exponential time for the ncSoS algorithm to witness this. By contrast, our algorithm runs in polynomial time.
3. Existence of an unsatisfiable phase for random (non-symmetric) XOR games. We show that there exists a constant  $C_k^{\text{unsat}}$  depending only on the number  $k$  of players, such that a random  $k$ -XOR game over an alphabet of size  $n$  has  $\omega^* < 1$  with high probability when the number of clauses is above  $C_k^{\text{unsat}} n$ .
4. A lower bound of  $\Omega(n \log(n) / \log \log(n))$  on the number of levels in the ncSoS hierarchy required to detect unsatisfiability for most random 3-XOR games. This is in contrast with the classical case where the  $(3n)^{\text{th}}$  level of the sum-of-squares hierarchy is equivalent to brute-force enumeration of all possible solutions.

**2012 ACM Subject Classification** Theory of computation → Quantum complexity theory

---

<sup>1</sup> ABW was supported by NSF grant CCF-1729369.

<sup>2</sup> AWH was funded by NSF grants CCF-1452616, CCF-1729369, ARO contract W911NF-17-1-0433 and the MIT-IBM Watson AI Lab under the project *Machine Learning in Hilbert space*.

<sup>3</sup> GK was supported by DOE grant DE-SC0011090.

<sup>4</sup> AN was supported by NSF grant CCF-1452616.



**Keywords and phrases** Nonlocal games, XOR Games, Pseudotelepathy games, Multipartite entanglement

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2019.10

**Related Version** A full version of the paper is available at <https://arxiv.org/abs/1801.00821>.

**Acknowledgements** The authors would like to thank Jöp Briet for helpful discussions, Oded Regev for pointing out the behavior of parallel-repeated GHZ games, anonymous reviewers for helpful comments on an earlier draft of this work, as well as George Wang, who contributed to this project as part of the RSI summer program at MIT. Parts of this work were completed while AWH and AN were hosted at the Institut Henri Poincaré in Paris, as part of the special semester on Analysis in Quantum Information Theory (Fall 2017), supported by NSF Grant DMS-1700168. The hospitality of the IHP is gratefully acknowledged.

## 1 Introduction

Constraint satisfaction problems (CSPs) are a fundamental object of study in theoretical computer science. In quantum information theory, there are two natural analogues of CSPs, which both play important roles: local Hamiltonians and (our focus) non-local games. Non-local games originate from Bell’s pioneering 1964 paper, which showed how to test for quantum entanglement in a device with which we can interact only via classical inputs and outputs. In modern language, the tests developed by Bell are games: a referee presents two or more players with classical questions drawn from some distribution and demands answers from them. Each combination of question and answers receives some score and the players cooperate (but may not communicate) to maximize their expected score. These games are interesting because often the players can win the game with a higher probability if they share an entangled quantum state, so a high average score can certify the presence of quantum entanglement. Such tests are not only of scientific interest, but have had wide application to proof systems [8, 19], quantum key distribution [1, 13, 29], delegated computation [26], and randomness generation [10], among others.

To be able to use a nonlocal game as a test for entanglement, it is essential to be able to approximately compute two quantities: the best possible expected score when the players share either classical correlations or entangled states, respectively called the “classical” and “quantum” (or “entangled”) values of the game, and denoted  $\omega$  and  $\omega^*$ . Classically, our understanding of the complexity of computing  $\omega$  rests on the intimate connection between games and CSPs. Indeed, there are several natural ways to map a CSP into a game. Perhaps the most commonly used is the “clause-variable game,” in which a CSP of any arity  $k$  is mapped to a two-player game, where one player is asked for the assignment to a clause of the CSP, and the other for the assignment to a single variable. However, there is another natural yet perhaps less-studied reduction that maps a  $k$ -ary CSP to a  $k$ -player nonlocal game, which moreover is symmetric under exchange of the players. In this reduction, given a CSP with a  $k$ -ary predicate, the referee of the game chooses uniformly at random a single clause, consisting of a  $k$ -tuple of variables and set of accepted assignments. The referee will then ask each of the  $k$  players for the value of one of the  $k$  variables in the clause, and accept if and only if the returned values constitute an accepted assignment to the clause. Classically, a simple convexity argument shows that the players can always stick to *deterministic* strategies, where each question is assigned a fixed answer, and for odd  $k$ ,

it is easy to show that there is a close relation between  $\omega$  and the CSP value: if the CSP has value 1 (i.e. all clauses are satisfiable),  $\omega = 1$ , and if its value is at most  $1 - \delta$ , then  $\omega \leq 1 - \delta/k$ . Hence, thanks to various dichotomy theorems, we have a good understanding of the difficulty of computing  $\omega$  for symmetric games<sup>5</sup>: in some cases, we know a P algorithm, and for most others, we know it is NP-complete. In particular, thanks to [18], this is known even for games where the referee’s acceptance depends only the XOR of single-bit answers from the players. Such games are known as XOR games.

The hardness of computing quantum value  $\omega^*$  is not as well understood, both in terms of upper and lower bounds. We know striking examples of quantum “advantage” (i.e. cases where the quantum value of a game is higher than the classical value), such as a Magic Square game, a game arising from an unsatisfiable CSP which nevertheless has an entangled strategy that succeeds with certainty, and thus quantum value  $\omega^* = 1$ . This advantage is also the main obstacle to our understanding, in that the set of entangled strategies is very rich: the “assignment” to each variable is no longer a value from a discrete set, but a linear operator over a Hilbert space of potentially unbounded dimension. Indeed, if infinite-dimensional entanglement is allowed, then depending on how one implements the requirement of non-communication between the players, one can obtain two different notions of entangled value – the tensor product value  $\omega_{TP}^*$  and the commuting operator value  $\omega_{CO}^*$  – which are not known to be equal.

As a result of the difficulties of unbounded-dimensional entanglement, we can say very little in terms of upper bounds on the complexity of computing either version of  $\omega^*$ , and in fact, it is not known whether even a constant-factor (additive) approximation to either is Turing-computable. For general games, the best we can say is they are recursively enumerable: for  $\omega_{TP}^*$ , there is a straightforward brute-force search over all strategies that in the limit of infinite time converges from below, and for  $\omega_{CO}^*$ , there is an algorithm, called the NPA or ncSoS hierarchy [22, 11], that in the limit of infinite time converges from above, but with no bound on the speed of convergence for either algorithm. On the hardness side, what we know is based on exploiting the CSP-game connection outlined above, but technically this has proved significantly more challenging than in the classical case. For instance, it was shown by Vidick that in the worst case, computing a constant-factor approximation to  $\omega_{TP}^*$  for 3-player XOR games is NP-hard [30], matching the classical hardness of [18], but this required redoing the soundness analysis of a PCP construction in the presence of entanglement. For general (non-XOR) games and tighter approximations we have super-classical hardness results [20, 21, 14]. Moreover, families of games with a “clause-variable” structure have been found for which deciding whether  $\omega^* = 1$  is uncomputable [27]. At the same time, we know that for certain families of games,  $\omega^*$  is easy to compute. Perhaps the best understood case is two-player XOR-games, for which Tsirelson showed that a simple semidefinite program (the lowest level of the ncSoS) exactly computes  $\omega_{CO}^* = \omega_{TP}^*$ , in contrast to the classical case where  $\omega$  for such games is NP-hard to approximate. A second family of games where results are known is XOR games with a maximum of two questions per player, but any number of players. Here there is a classification of all correlations achievable by quantum players, as well as a description of the measurement strategy players use to achieve these correlations. Interestingly, we arrive at the same measurement strategy later in this work through independent techniques.

---

<sup>5</sup> Classically, there are simple reductions from the general case to the symmetric case, but as we discuss below, these fail to preserve completeness in the presence of entanglement.

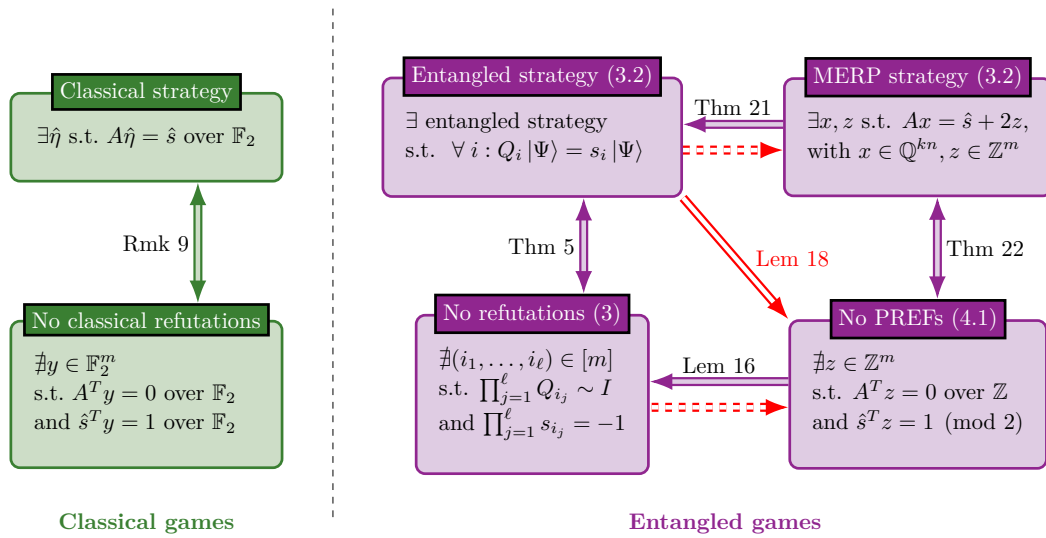
From the preceding results, XOR games emerge as a natural class of games to understand on the road to a full “dichotomy theorem” for quantum games. Classically, XOR games are also convenient to analyze because of their linear structure: a  $k$ -player XOR game represents a CSP whose clauses are linear equations over the finite field  $\mathbb{F}_2$ , each containing  $k$  variables. As a result, classically XOR games are always easy in the “perfect completeness” regime: we can determine whether an XOR game is perfectly satisfiable in polynomial time using Gaussian elimination over  $\mathbb{F}_2$ , even though distinguishing  $\geq 1 - \varepsilon$  satisfiability from  $\leq \frac{1}{2} + \varepsilon$  satisfiability is NP-complete. This linear structure also makes it easy to reason about the classical value of random instances of XOR games using linear algebra. However, this simple linear structure does not capture entangled strategies and neither the Gaussian elimination algorithm for the perfect completeness regime, nor the classical analysis of random instances generalizes easily to the quantum case. Indeed, the undecidability result of [27] applies to the perfect completeness regime for games based on systems of linear equations, though these systems are not over  $\mathbb{F}_2$  and the games are in the two-player “clause-variable” format. Is the perfect completeness regime for quantum XOR games easy, as in the classical case, or hard, as suggested by Slofstra’s results? And what can we say about random instances?

In this work, we make progress on these questions for the subclass of *symmetric* XOR games: those for which the game remains invariant under any permutation of the players. This class of games includes those arising from CSPs via the reduction described above, as well as the hard instances of [30]. Our main results are captured by the following theorem.

► **Theorem 1** (Theorems 14 and 15 in the body). *A symmetric  $k$ -player XOR game has entangled value  $\omega_{CO}^* = \omega_{TP}^* = 1$  if and only if an associated system of linear Diophantine equations has no solution. This condition can be checked in polynomial time, and whenever it is satisfied, the perfect tensor-product strategy can be found in polynomial time. When it is not satisfied, a succinct description of an ncSoS dual certificate that  $\omega^* < 1$  can be found in polynomial time (even though the certificate may be exponentially long).*

We achieve these results by viewing an XOR game as a “non-commuting” generalization of linear systems of equations, in which the expectation of differences between products of operator-valued variables and plus or minus the identity operator are constrained to be zero. We develop a “duality theory” for these systems of operator equations, where the dual certificates of infeasibility correspond to a special class of ncSoS proofs which we call “refutations.” For symmetric games, we show that a dual certificate exists if and only if a certain system of linear Diophantine equations has a solution (which we call a “PREF”). An important feature of our algorithm is that while it is inspired by ncSoS, its performance can be significantly superior: it can detect in polynomial time the existence of an exponentially long ncSoS dual certificate. Indeed, we show (in Theorem 23) a concrete family of games where our algorithm can detect that  $\omega^* < 1$  in time which scales polynomially in the game size  $n$ , whereas ncSoS takes doubly exponential time. We believe this result may be interesting in its own right to those who study the Sum of Squares algorithm, and hope that our techniques inspire further efficient algorithms that “simulate” high levels of SoS. Additionally, by further considering the dual of the system of Diophantine equations we construct, we are able to extract a simple finite-dimensional (and hence tensor product) strategy (which we call “MERP”) that achieves  $\omega_{TP}^* = \omega_{CO}^* = 1$  whenever the system of equations has a solution. A diagram illustrating the dualities we use is given in Figure 1.

Our notion of refutation is similar to the “substitution method” in the prior work of [9], used there to analyze clause-variable style games (there called Binary Constraint System Games) in the perfect-completeness regime. However, the connection we show between refutations and linear Diophantine equations, which is the heart of our efficient algorithm for



**Figure 1** We extend the well-understood duality relation for classical XOR games (left) to a more complex set of dualities characterizing perfect strategies for entangled XOR games (right). The arrows indicate implications, with the red, unfilled arrows holding for symmetric games only. The dashed red arrows follow from the key lemma for symmetric games. Definitions and notation are developed in the remaining sections.

searching over refutations, is new to this work and makes essential use of the properties of symmetric XOR games. We consider it an interesting open question whether our techniques could be adapted to the Binary Constraint System case.

The symmetry condition on the game is important to our analysis, and it is worth going into some detail as it presents an interesting divergence from the classical case. Classically, any game can be symmetrized as follows: for each clause consisting of questions  $(q^{(1)}, \dots, q^{(k)})$  asked to players  $1, \dots, k$ , the referee chooses a random permutation  $\pi$  of  $\{1, \dots, k\}$ , and sends player  $i$  the pair  $(\pi(i), q^{(\pi(i))})$ . Each player  $i$  then follows the same strategy that player  $\pi(i)$  would have used in the original, unsymmetrized game. In the quantum setting, this transformation fails to preserve completeness: for instance, if an entangled strategy for a three-player unsymmetrized game requires players 1 and 2 to share entanglement, in the symmetrized version, a player receiving the index 1 does not know which other player received index 2, and thus does not know who to be entangled with. This can be understood as an instance of the phenomenon of *monogamy*, which distinguishes entanglement from classical correlations. It is an interesting question for future work to extend our methods to the nonsymmetric case.

Furthermore, as alluded to earlier, our algorithm yields an understanding of the typical value of a random symmetric XOR game. Classically, research in this direction draws significantly on insights from statistical mechanics and has proven that there exist sharp satisfiable/unsatisfiable thresholds for random  $k$ -SAT and related games. But these techniques do not carry over to the quantum case. For random classical games, a basic method is to look at the expected number of winning strategies (the “first moment method”) or the variance (the “second moment method”) as we randomize the referee’s payoff function within some family such as random  $k$ -SAT or random  $k$ -XOR. This suffices, for example, to show that random 3-XOR games with  $n$  variables and  $Cn$  clauses are satisfiable with high probability if and only if  $C \lesssim 0.92$  [12]. Since quantum strategies do not form a discrete (or even finite-dimensional)

set, these methods are not possible. Nor is it obvious how to use more refined tools such as Shearer’s Lemma or the Lovász Local Lemma, which address the question of when sets of overlapping constraints can be simultaneously satisfied. Our duality theory enables us to avoid these obstacles by studying refutations, rather than strategies. Refutations are discrete objects and thus are more amenable to combinatorial and probabilistic techniques. Using our techniques we are able to prove that random quantum XOR games have an unsatisfiable phase above a certain clause density.

► **Theorem 2** (Theorem 26 in the body). *For every  $k$ , there exists a constant  $C_k^{unsat}$  depending only on  $k$  such that a random  $k$ -XOR game  $G$  with  $m \geq C_k^{unsat}n$  clauses has value  $\omega^*(G) < 1$  with probability  $1 - o(1)$ .*

In our overall approach in this paper, we were inspired by the work of Grigoriev [17], who studied the power of SoS refutations for random classical XOR games. We view Theorem 2, together with the ncSoS lower bounds of Theorem 23, as a quantum generalization of Grigoriev’s results.

## 2 XOR Games

We begin by defining a  $k$ -XOR game, along with its classical and quantum values.

► **Definition 3.** Define a **clause**  $c = (q, s)$  to be any  $(k + 1)$ -tuple consisting of a **query**  $q \in [n]^k$  and **parity bit**  $s \in \{-1, 1\}$ .

In a  $k$ -XOR game  $G$  associated with a set of clauses  $M$ , a verifier selects a clause  $c_i = (q_i, s_i)$  uniformly at random from  $M$ . For all  $\alpha \in [k]$ , the **question**  $q_i^{(\alpha)}$  is then sent to the  $\alpha$ -th player of the game. Without communicating, the players then each send back a single output  $\in \{-1, 1\}$ , and win the game if their outputs multiply to  $s_i$ .

The GHZ game [16] is a canonical example of a 3-XOR game. It is defined by the clauses (here we use the labels  $\{x, y\}$  for the questions instead of the typical  $\{1, 2\}$ ):

$$G_{GHZ} := \left\{ \begin{array}{l} \left[ \begin{array}{c} x \\ x \\ x \\ +1 \end{array} \right], \left[ \begin{array}{c} y \\ y \\ x \\ -1 \end{array} \right], \left[ \begin{array}{c} y \\ x \\ y \\ -1 \end{array} \right], \left[ \begin{array}{c} x \\ y \\ y \\ -1 \end{array} \right] \end{array} \right\} \begin{array}{l} \leftarrow \text{Player A} \\ \leftarrow \text{Player B} \\ \leftarrow \text{Player C} \\ \leftarrow \text{Desired product} \end{array} \quad . \quad (1)$$

There is a natural reduction from a  $k$ -CSP over  $\mathbb{F}_2$  to a  $k$ -XOR game, based on the isomorphism between the groups  $(\{0, 1\}, +_{\text{mod } 2})$  and  $(\{1, -1\}, \times)$ . The XOR game corresponding to a CSP has clauses defined by picking a clause from the CSP at random, sending each player a question corresponding to a random distinct variable from the CSP clause, then choosing a parity bit by demanding that players’ answers satisfy the CSP. Games constructed from CSPs in this manner are symmetric over permutation of the players, and are therefore called **symmetric games**.

By excluding communication during the game, the classical game tests whether the players have cooperatively solved the CSP before the questioning began. When the players are given access to quantum resources, the game instead probes “quantum solutions” to the CSP, described by measurements of a shared state in some Hilbert space.

The **value** of that game is defined to be the optimal win probability obtained by the players. We distinguish various possible classes of resources that may be made available to the players in executing a strategy, each of which defines a particular type of value.

► **Definition 4.** We define three versions of the value of game  $G$ .

1. The **classical value**  $\omega(G)$  is the value achievable by players using only classical shared information.
2. The **tensor-product value** is the value obtainable by players sharing a quantum state but restricted to making measurements on distinct factors of a tensor-product Hilbert space. Intuitively, this is a no-communication condition described in Hilbert-space language.
3. The **commuting-operator value**<sup>6</sup>  $\omega^*(G)$  is the value obtainable by players making commuting measurements on a shared quantum state. Intuitively, this is a weaker form of the no-communication constraint, permitting states living in non-separable Hilbert spaces.

When a CSP is reduced to an XOR game  $G$ , the classical value roughly corresponds to the fraction of satisfiable constraints, with  $\omega(G) = 1$  if and only if the CSP is satisfiable. Sharing a quantum state may allow the players to provide a quantum solution ( $\omega^* = 1$ ) even when  $\omega \neq 1$ . Famously, the GHZ game is a symmetric game that corresponds to a test of the classically-unsatisfiable, yet quantumly-soluble CSP:

$$x + x + x = 0 \pmod{2} \quad \text{and} \quad x + y + y = 1 \pmod{2}. \quad (2)$$

Games (such as the GHZ game) that satisfy  $\omega < 1$  and  $\omega^* = 1$  are called **pseudo-telepathy games**. Identifying other XOR pseudo-telepathy games is one of the motivating goals of this work.

For a given game, the set of values achievable by tensor-product strategies may not be closed [27]. Whether the closure of this set can differ from the commuting-operator value of the game remains unanswered<sup>7</sup>. In this paper, we focus primarily on a description of the commuting-operator value but in many cases can show that it coincides with the tensor-product value.

### 3 Refutations

A main aim of this paper is to characterize the set of XOR games with commuting-operator value  $\omega^* = 1$ . In the case of  $k = 2$  players, Tsirelson gave an efficient semidefinite program that computes the exact value of  $\omega^*$ ; however, this technique does not generalize easily to  $k \geq 3$  [6, 28]. Furthermore, the potentially unbounded size of the players' resource state makes it impossible to upper bound the value of a game via brute force search over strategies.

To avoid these problems, this work introduces a dual characterization that certifies games with value  $\omega^* < 1$ . This is a natural generalization of a well-understood dual system of equations that certifies games with classical value  $\omega < 1$ , and employs operator language similar to the quantum satisfying assignments for Binary Constraint System games presented in [9]. The dual pictures in both the classical and commuting-operator cases introduce the notion of a “refutation”: intuitively, a sequence of game clauses that together contradict the existence of a value-1 strategy. We show Theorem 5 that refutations are dual to  $\omega^* = 1$ ; our proof can be viewed as a quantum generalization of [17].

<sup>6</sup>  $\omega^*(G)$  is often also referred to as the field-theoretic value of  $G$ .

<sup>7</sup> And hard! For general two-player games this question is known to be equivalent to Connes' embedding conjecture [15].



► **Theorem 5** (Strategy-Refutation Duality). *An XOR game  $G$  has commuting-operator value  $\omega^*(G) = 1$  if and only if it admits no refutations.*

We first take a small detour into the classical duality picture to build intuition and necessary notation (Section 3.1), then describe refutations and outline the proof of duality in the quantum case (Section 3.2).

### 3.1 Classical Strategies and Refutations

Classically, refutations emerge naturally from the linear-algebraic dual to the equations satisfied by a classical value-1 strategy.

For any game, the optimal classical strategy can be specified via a map  $[kn] \rightarrow \{1, -1\}$  giving a deterministic answer to each possible question given to each player. In order to use linear algebraic tools, we exploit the isomorphism  $(\{1, -1\}, \times) \sim (\{0, 1\}, +_{\text{mod } 2})$  and specify a classical strategy via a vector  $\hat{\eta} \in \mathbb{F}_2^{kn}$ . Explicitly:  $\hat{\eta}_{(\alpha-1)n+j} = 0$  if player  $\alpha$  responds to question  $j$  with a 1, and equals 1 if the player responds with a  $-1$ . To clearly specify the player and question, we use the notation

$$\hat{\eta}(\alpha, j) := \hat{\eta}_{(\alpha-1)n+j}.$$

From this point on, we will freely switch back and forth between an additive and a multiplicative representation of strategies, leaving the mapping implicit.

To complete the linear algebraic picture, we also define a vector of desired outputs  $\hat{s}$  and a game matrix  $A$ . The game matrix is defined such that given a strategy vector  $\hat{\eta}$ , the parities of the outputs for each clause are given by the vector  $A\hat{\eta}$ .

► **Definition 6.** Given a  $k$ -XOR game with  $m$  queries and alphabet size  $n$ , the **game matrix**  $A$  is an  $m \times kn$  matrix describing query-player-question incidence, and the length- $m$  **parity bit vector**  $\hat{s} \in \mathbb{F}_2^m$  the desired outputs:

$$A_{i,(\alpha-1)n+j} := \begin{cases} 1 & \text{if } q_i^{(\alpha)} = j \\ 0 & \text{otherwise} \end{cases} \quad \text{and} \quad \hat{s}_i := \begin{cases} 0 & \text{if } s_i = 1 \\ 1 & \text{if } s_i = -1 \end{cases}. \quad (3)$$

An XOR game  $G$  is completely specified by providing the game matrix  $A$  and parity bit vector  $\hat{s}$ , i.e.  $G \sim (A, \hat{s})$ . For example, we translate the GHZ queries into  $A_{GHZ}$  and parity bits into  $\hat{s}_{GHZ}$  by:

$$\implies A_{GHZ} := \left( \begin{array}{c|c|c} (A) & (B) & (C) \\ \hline 1 & 0 & 1 \\ 0 & 1 & 0 \\ 0 & 1 & 1 \\ 1 & 0 & 0 \end{array} \right) \quad \text{and} \quad \hat{s}_{GHZ} := \begin{pmatrix} 0 \\ 1 \\ 1 \\ 1 \end{pmatrix}. \quad (4)$$

A linear-algebraic constraint for achieving classical value 1 can then be defined by asking that a strategy exists that outputs the desired  $\hat{s}$ .

► **Definition 7.** The **classical constraint equation** for strategy  $\hat{\eta}$  on game  $G \sim (A, \hat{s})$  is

$$A\hat{\eta} = \hat{s} \quad (\text{over } \mathbb{F}_2). \quad (5)$$

The solutions to (5) are exactly the classical strategies achieving value 1 on game  $G \sim (A, \hat{s})$ . In other words, a game  $G$  has classical value 1 iff (5) has a solution. Gaussian elimination can be used to check for a solution to (5), so we can decide whether a game has  $\omega = 1$  in P. Even so, transforming to the dual picture provides a useful analog to the quantum case.



► **Definition 8.** Define a **classical refutation**  $y \in \mathbb{F}_2^m$  as any vector satisfying the equation dual to (5),

$$\begin{bmatrix} A^T \\ \hat{s}^T \end{bmatrix} y = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \quad (\text{over } \mathbb{F}_2). \quad (6)$$

► **Theorem 9.** *Either a classical refutation  $y$  satisfying (6) or a classical strategy  $\hat{\eta}$  satisfying (5) must exist.*

**Proof.** Immediate from the observation that equations (5) and (6) are dual. ◀

A refutation  $y$  has a direct interpretation as a certificate that  $\omega < 1$ : collecting constraints from clauses  $i$  corresponding to non-zero entries  $y_i$  produces a contradiction to the value-1 hypothesis. To understand this, note that satisfying clause  $i$  requires that the  $i$ th row of (5) is true under strategy  $\hat{\eta}$ ,

$$\begin{aligned} [A\hat{\eta}]_i &= \hat{s}_i \\ \Leftrightarrow \sum_{\alpha} \hat{\eta}(\alpha, q_i^{(\alpha)}) &= \hat{s}_i \pmod{2}. \end{aligned}$$

Summing the value-1 constraints selected by  $y$  (left-multiplication by  $y$ ) produces the desired contradiction when  $y$  satisfies (6),

$$\begin{aligned} \sum_{i:y_i=1} \sum_{\alpha} \hat{\eta}(\alpha, q_i^{(\alpha)}) &= \sum_{i:y_i=1} \hat{s}_i \\ \Leftrightarrow y^T A\hat{\eta} &= y^T \hat{s} \pmod{2} \\ \implies 0 &= 1 \pmod{2}. \end{aligned}$$

This interpretation is key to generalizing refutations to the commuting-operator value of XOR games.

### 3.2 Commuting-Operator Strategies and Refutations

Whereas classical strategies are specified by assigning deterministic output to every player-question pair, commuting-operator strategies are specified by assigning a  $\pm 1$  valued quantum measurement to every player-question pair and fixing some entangled state shared by the players. Each player then executes a commuting-operator strategy by selecting the measurement corresponding to the question they receive, then returning the result of applying it to the shared state.

Using the Naimark dilation theorem, we can restrict the measurements in the players' strategies to be Projection-Valued Measures (PVMs). This is the quantum mechanical analogue of the statement that the optimal classical strategy can be taken to be deterministic. In the case of XOR games, this means the measurements can be chosen to be a pair of projectors  $\{O_1, O_{-1}\}$  that partition the space into two subspaces, corresponding to outputs 1 and  $-1$ . We make this restriction for the remainder of the paper.

All that remains is to enforce the no communication requirement on the quantum players. This is done in one of two possible ways. In tensor product strategies the Hilbert space in which  $|\Psi\rangle$  lives is taken to be separable, with different players' measurements acting on disjoint parts of the state. In commuting-operator strategies no restriction is placed on the Hilbert space, but the Hermitian matrices corresponding to different players' measurements are forced to commute. (These two restrictions are distinct only in the case of an infinite dimensional Hilbert space). In this paper we work exclusively with the commuting-operator definition, though all the explicit strategies we construct are also valid tensor product strategies.

## 10:10 Algorithms, Bounds, and Strategies for Entangled XOR Games

Putting this all together, we can define for any strategy the observables corresponding to the players' measurements.

► **Definition 10.** Given a commuting-operator strategy consisting of measurements  $\{O_1^\alpha(j), O_{-1}^\alpha(j)\}$  for each possible question  $j$  and player  $\alpha$ , define the Hermitian **strategy observable** for player  $\alpha$  upon receiving question  $j$

$$O^\alpha(j) := O_1^\alpha(j) - O_{-1}^\alpha(j). \quad (7)$$

Operators  $O^\alpha(j)$  can equivalently be chosen without reference to particular PVMs by taking any set of Hermitian operators that satisfy the constraints (for all players  $\alpha \neq \beta$  and questions  $j, j'$ )

$$[O^\alpha(j), O^\beta(j')] = 0 \quad (\text{operators held by distinct players commute}) \quad (8a)$$

$$(O^\alpha(j))^2 = I \quad (\text{square identity, enforcing } \pm 1 \text{ eigenvalues}). \quad (8b)$$

This abstract definition of strategy observables will be the one most frequently referenced in the remainder of this paper.

Given Hermitian observables, the condition for commuting-operator strategies to achieve value 1 is an eigenvalue condition, generalizing (5).

► **Definition 11.** For a  $k$ -XOR game  $G$ , define the **commuting-operator constraint equations**:

$$\forall i \in [m] : \quad Q_i |\Psi\rangle = s_i |\Psi\rangle \quad (9)$$

where the **query observable**  $Q_i := \prod_\alpha O^\alpha(q_i^{(\alpha)})$  is the product of all players' observables for the  $i$ th query.

A strategy achieving value  $\omega^* = 1$  must be played on a state  $|\Psi\rangle$  which is an eigenvector of every query observable, with appropriate eigenvalue, to ensure zero probability of outputting an incorrect response to some query. This eigenvalue criterion guarantees that the players win all queries. A game  $G$  therefore has commuting operator value  $\omega^* = 1$  iff there exists some state and strategy observables that satisfy (8) and (9).

While there is an efficient algorithm to solve the classical constraint equations, no such algorithm is known to exist for the commuting-operator constraint equations. Indeed, there is no known upper bound on the dimension of the Hilbert space required to optimally play an entangled game, meaning the search space of the commuting-operator equations is not finite, and the equations themselves may be undecidable. To work around this we develop refutations to characterize the commuting-operator value of XOR games. This technique gives a search space which is still infinite, but is at least discrete, allowing for some progress to be made via combinatorial analysis.

We would like to construct a dual to the commuting-operator constraint equations, meaning a certificate for the unsatisfiability of (9). As there is no immediate analogue to the linear algebraic methods used in the classical case, we instead generalize the view of a refutation as a collection of clauses producing a contradiction (similar to (6)). At a high level, refutations are obtained by multiplying together constraints of the form (9) and applying the known operator identities of (8a) and (8b) to arrive at an equation of the form  $I |\Psi\rangle = -|\Psi\rangle$  which cannot be true for a normalized quantum state.<sup>8</sup>

<sup>8</sup> Importantly, the order in which the constraint equations are multiplied matters, as two distinct commuting-operator strategy observables with the same player label may not commute. Further, the same constraint equation may need to be incorporated multiple times before one can arrive at a contradiction.

To formally define a refutation, we use an equivalence relation between possible strings of strategy observables on the LHS of an equation of the form (9).

► **Definition 12.** Let  $Z_1$  and  $Z_2$  be two operators formed from products of strategy observables. We say  $Z_1$  is equivalent to  $Z_2$ , written  $Z_1 \sim Z_2$ , if  $Z_1 = Z_2$  is an identity for all strategy observables satisfying (8).

Definitions 11 and 12 then allow us to precisely define a (quantum) refutation, analogous to Definition 8. From now on, a “refutation” will be a quantum refutation unless otherwise specified.

► **Definition 13.** Let  $G$  be some  $k$ -XOR game with  $m$  clauses. A **refutation for  $G$**  is defined to be a sequence of clause indices  $(i_1, i_2, \dots, i_\ell) \in [m]^\ell$  satisfying

$$Q_{i_1} Q_{i_2} \dots Q_{i_\ell} \sim I \quad \text{and} \quad s_{i_1} s_{i_2} \dots s_{i_\ell} = -1. \quad (10)$$

Assuming the value-1 hypothesis and combining the  $\ell$  constraint equations satisfying (10) then gives the desired contradiction,  $I|\Psi\rangle = -|\Psi\rangle$ . Whether a product of queries  $Q_{i_1} Q_{i_2} \dots Q_{i_\ell}$  is equivalent to  $I$  can be efficiently checked by collecting each player’s operators using (8a) and repeatedly applying (8b) to greedily cancel operators.

Refutations certify that  $\omega^* < 1$  analogously to the way that classical refutations certify that  $\omega < 1$ . We prove in the full paper that the converse is also true, completing the proof of Theorem 5. The proof of this fact relies on a connection between refutations and the ncSoS hierarchy analogous to a connection made by Grigoriev [17] between classical refutations and the SoS hierarchy. In particular, we show the ncSoS algorithm takes time exponential in the minimum length refutation to prove a game has value  $\omega^* < 1$ . Theorem 5 then follows from completeness of ncSoS.

It is not obvious that one can find refutations more easily than one can find strategies. The remaining results focus primarily on subclasses of XOR games for which we can apply the refutations picture to exactly characterize the games with  $\omega^* = 1$ . In particular, we identify an easily-computed stronger refutation condition that is complete for symmetric games, i.e. those naturally corresponding to CSPs. Subsequently we analyze specific families of games that give bounds on the behavior of ncSoS and insight into the structure of the XOR game landscape.

## 4 Symmetric Games

The refutation technology developed above gives surprisingly powerful results when applied to symmetric games. In particular, we show:

► **Theorem 14.** *Membership in the set of symmetric games with  $\omega^* = 1$  can be efficiently decided via a system of linear Diophantine equations.*

Previously the question of whether symmetric games took value 1 was not known to be decidable. Theorem 14 affirms that it is decidable and in fact in P. We prove the theorem by introducing a simple necessary condition for refutations to exist, then showing the structure of symmetric games ensures this condition is also sufficient for a refutation.

Games that do not satisfy this necessary condition have value  $\omega^* = 1$ . By returning to duality arguments, we further show that they can be played optimally by a simple family of strategies.

► **Theorem 15.** *Any value-1 symmetric game can be played optimally by a single qubit strategy using a GHZ resource state. Furthermore, this strategy can be found in polynomial time.*

#### 4.1 A Necessary Condition for Refutation (PREF)

Any valid refutation involves a product of query observables that cancel to  $I$  via the square identity (8b). Focusing on the strategy observables corresponding to one player, we see every operator at an even depth in the sequence must cancel with one at an odd depth. Given any sequence of query observables we can count the number of copies of  $O^\alpha(j)$  at odd and even depths – if the sequence corresponds to a refutation the counts must be equal. Then for any given game, it is necessary to be able to construct some sequence of queries  $Q_{i_1} \dots Q_{i_\ell}$  satisfying this counting equality (with appropriate parity  $s_{i_1} \dots s_{i_\ell} = -1$ ) in order to construct a true refutation. We call such a sequence a **parity-permuted refutation (PREF)**.

To prove properties of such PREFs, we find it useful to introduce a freer equivalence relation  $\overset{\mathcal{L}}{\sim}$  on strings of queries that allows reordering within the even positions and the odd positions before cancellation (compare to Definition 12). A PREF is then a string of clauses  $(i_1, \dots, i_\ell)$  satisfying  $Q_{i_1} \dots Q_{i_\ell} \overset{\mathcal{L}}{\sim} I$  and the same parity-bit requirement as a regular refutation. In later sections, we carefully define and use a technical version of  $\overset{\mathcal{L}}{\sim}$ , but here simply state that  $\overset{\mathcal{L}}{\sim}$  is a more inclusive equivalence relation than  $\sim$ , which immediately gives Lemma 16.

► **Lemma 16** (Necessary condition for refutation). *If a game  $G$  admits a refutation, it contains a PREF.*

We define **noPREF games** to be those games that do not admit a PREF. These games admit no true refutations by the previous lemma, and so have  $\omega^* = 1$ . Whether or not a game contains a PREF is efficiently decidable by checking for a solution to a linear system of equations. We sketch the algorithm that decides membership, delegating a rigorous proof to the full paper.

► **Lemma 17** (Informal). *Membership in the set of noPREF games can be efficiently decided by a system of linear Diophantine equations.*

**Proof (sketch).** We prove the result by showing that a game  $G \sim (A, \hat{s})$  contains a PREF if and only if there is a solution to the set of equations

$$A^T z = 0 \tag{11}$$

$$\hat{s}^T z = 1 \pmod{2} \tag{12}$$

for some  $z \in \mathbb{Z}^m$ . To prove the forward direction we note that each row of (11) guarantees that a particular player-question pair has equal positive and negative count and (12) ensures the parity bit requirement is met, such that the game  $G$  contains a PREF built by interleaving the multisets of clause indices

$$\mathcal{O} = \{i \text{ with multiplicity } |z_i| \mid \forall i : z_i > 0\} \tag{13a}$$

$$\mathcal{E} = \{i \text{ with multiplicity } |z_i| \mid \forall i : z_i < 0\} \tag{13b}$$

with their elements placed in odd and even positions, respectively. The reverse direction is proved similarly, but requires a technical lemma relating the even and odd clauses of a PREF. Then standard techniques for solving linear Diophantine equations complete the proof. ◀

Symmetric games have additional structure that allows us to prove a stronger statement.

► **Lemma 18** (Informal). *The noPREF characterization is complete for symmetric games. That is, every value 1 symmetric game is in the noPREF set.*

**Proof.** The proof of Lemma 18 is both constructive and purely combinatorial. It involves first showing that symmetric games have enough structure to construct “shuffle gadgets” which let us approximately commute strategy observables past each other. Then, careful application of these shuffle gadgets lets us transform a PREF into a true refutation. The technical proof is presented in the full paper. ◀

Theorem 14 then follows directly from Lemmas 17 and 18 since symmetric games have commuting-operator value 1 if and only if they admit no PREFs and this condition is efficiently checkable.

## 4.2 A Single Qubit Strategy (MERP)

We arrived at the noPREF classification of XOR games by generalizing the classical dual picture to the entangled games case, then finding a simple necessary condition for a dual proof (refutation) to exist. In this section we take the dual of a dual, and give a simple technique to construct a strategy that achieves value 1 for any noPREF game, thus sketching the proof of Theorem 15.

We call the resulting family of strategies **Maximal Entanglement, Relative Phase (MERP)** strategies. These strategies are exactly the class of strategies developed in [31] to optimally solve all games with alphabet size two. They are played on a  $k$  qubit  $GHZ$  state (one qubit per player), and achieve value one if and only if (Theorem 21) there is a solution to the equation

$$A\hat{\theta} = \hat{s} \pmod{2}, \quad \hat{\theta} \in \mathbb{Q}^{kn}.$$

► **Definition 19 (MERP).** Given a  $k$ -XOR game  $G$ , a **MERP strategy** for  $G$  is a tensor-product strategy in which:

1. The  $k$  players share the maximally entangled state

$$|\Psi\rangle = \frac{1}{\sqrt{2}} \left[ |0\rangle^{\otimes k} + |1\rangle^{\otimes k} \right] \tag{14}$$

with player  $\alpha$  having access to the  $\alpha$ -th qubit of the state.

2. Upon receiving question  $j$  from the verifier, player  $\alpha$  rotates his qubit by an angle  $\theta(\alpha, j)$  about the  $Z$  axis, then measures his qubit in the  $X$  basis and sends his observed outcome to the verifier.

Equivalently, we define the states

$$|\theta(\alpha, j)_{\pm}\rangle := \frac{1}{\sqrt{2}} \left[ |0\rangle \pm e^{i\theta(\alpha, j)} |1\rangle \right] \tag{15}$$

and pick strategy observables

$$O^{\alpha}(j) := |\theta(\alpha, j)_{+}\rangle\langle\theta(\alpha, j)_{+}| - |\theta(\alpha, j)_{-}\rangle\langle\theta(\alpha, j)_{-}|. \tag{16}$$

Each  $Z$  rotation executed by the players introduces a relative phase between the  $|0\rangle^{\otimes k}$  and  $|1\rangle^{\otimes k}$  components of the GHZ state, and these relative phases add. The measurement is constructed such that a total relative phase that is an even multiple of  $\pi$  results in overall output 1 while an odd multiple of  $\pi$  results in overall output  $-1$ . Collecting the rotation angles into a strategy vector

$$\hat{\theta}_{(\alpha-1)n+j} := \frac{1}{\pi} \theta(\alpha, j) \tag{17}$$

results in a useful parallel between MERP strategies and classical strategies:

► **Definition 20.** Define the **MERP constraint equations** for game  $G$  by

$$A\hat{\theta} = \hat{s} \pmod{2} \quad (18)$$

► **Theorem 21.** *The value obtained by a MERP strategy is given by*

$$v^{MERP}(G, \hat{\theta}) := \frac{1}{2} + \frac{1}{2m} \left( \sum_{i=1}^m \cos\left(\pi \left[ (A\hat{\theta})_i - \hat{s}_i \right] \right) \right). \quad (19)$$

Consequently, there exists a MERP strategy achieving  $v^{MERP} = 1$  on a game  $G$  iff its MERP constraint equations have a solution over  $\mathbb{Q}$ .

The proof of Theorem 21 is computational. For any game, Theorem 21 indicates that a MERP strategy achieves value 1 if and only if the MERP constraint equation (18) is satisfied. Similarly to the classical case, this can be efficiently checked by Gaussian elimination. Here, however, the underlying field is  $\mathbb{Q}$  as opposed to  $\mathbb{F}_2$ .

Both MERP strategies and PREF specifications are defined by linear systems of equations, over  $\mathbb{Q}$  and  $\mathbb{Z}$  respectively. Remarkably, these systems of equations are dual to each other, in much the same way as classical strategies and refutations. By showing this, we prove Theorem 22.

► **Theorem 22.** *Any game  $G$  admits either a PREF or a MERP strategy with value 1.*

**Proof.** Technical proof in the style of a Theorem of Alternatives presented in the full paper. ◀

Intuitively, this result (coupled with Lemma 18 and Theorem 21) indicates that the power of quantum solutions to noPREF games is equivalent to promoting the underlying field from  $\mathbb{F}_2$  to  $\mathbb{Q}$ . We expect further advantage to be gained from the non-commuting nature of operator-valued solutions in cases where a game admits a PREF but no true refutation. These classes of games are the main subject of future work.

Figure 1 (given in the Introduction) summarizes the new duality relations presented in this paper. We repeat them here. The general quantum duality provides a complex but complete description of games with  $\omega^* = 1$ . The PREF conditions are efficient to compute, but are only *necessary* conditions for constructing commuting-operator refutations, and thus the dual, MERP value 1, holds true for only a subset of all  $\omega^* = 1$  games. We can make a stronger statement about symmetric games: PREFs are both necessary and sufficient for a symmetric game to have a refutation, so the duality ensures MERP achieves value 1 for all symmetric games with  $\omega^* = 1$ .

## 5 ncSoS Bounds and the XOR Landscape

The refutations picture also allows us to give worst and average case bounds on the behavior of ncSoS for XOR games, and construct new families of games with interesting properties.

First, we construct a family of games that have  $\omega^* < 1$ , but are built in such a way that the ncSoS algorithm has a hard time recognizing this. Called Capped GHZ games, games in this family are symmetric and contain PREFs that all are at least exponentially long. The ncSoS algorithm<sup>9</sup> then requires time doubly exponential to prove that these games

<sup>9</sup> Our results imply that in the case of entangled XOR games the ncSoS has runtime within a polynomial factor of a brute force search over refutations up to a certain length.

have commuting-operator value  $< 1$ . This gives a rare example of a problem whose solution requires a superlinear number of levels of ncSoS, and illustrates the distinction between ncSoS and SoS, the latter always terminating after at most a linear number of levels. On the other hand, Lemma 18 tells us the existence of a PREF for this symmetric game indicates  $\omega^* < 1$ , which allows us to solve the same problem in polynomial time. This leads us to prove:

► **Theorem 23.** *There exists a family of 3-XOR games with  $\omega^* < 1$  but for which the minimum refutation length scales exponentially in the number of clauses  $m$  and alphabet size  $n$ . For these games exponentially many levels of ncSoS are needed to witness that  $\omega^* < 1$ .*

**Proof.** An explicit construction of Capped GHZ games presented in the full paper. These are symmetric games designed such that the PREF condition for these games can only be satisfied by sets containing exponentially many clauses. Since any refutation also gives a PREF, this means all refutations have at least exponential length. Finally, as a symmetric game, the existence of a PREF indicates the game has  $\omega^* < 1$ . By the connection between ncSoS and refutations, this means the ncSoS algorithm requires at least exponentially many levels to detect the game has value less than one. ◀

Returning to pseudo-telepathy, we construct a family of games that generalize the GHZ game, termed the Asymptotically Perfect Difference (APD) family. Members are parameterized by scale  $K$ , with the  $K$ -th member having  $k = 2^K - 1$  players. The APD family is designed such that any desired parity bits  $s_i$  can be produced by some strategy (the game is in the noPREF set regardless of the  $s_i$ , so  $\omega^* = 1$  for all  $s_i$ ). On the other hand, a growing fraction of possible assignments of  $s_i$  correspond to low classical value, and the family has perfect difference [4] in the asymptotic limit,  $\lim_{K \rightarrow \infty} 2(\omega^* - \omega) = 1$ . In comparison, there are randomized constructions of families of games whose bias ratio  $\frac{\omega^* - 1/2}{\omega - 1/2}$  diverges for fixed  $k \geq 3$  as  $n \rightarrow \infty$  [25, 4], but these constructions give no guarantee on the difference. Specifically, we prove it is possible to force an upper bound on  $\omega$  in terms of the number of players  $k$  while preserving  $\omega^* = 1$ :

► **Theorem 24.** *There exists a family of  $k$ -XOR games, parametrized by  $K$ , for which  $\omega^*(G(K)) = 1$  and the classical value is bounded by*

$$\frac{1}{2} \leq \omega(G(K)) \leq \frac{1}{2} + \sqrt{\frac{K+1}{2^{K+1}}} \leq \frac{1}{2} + \sqrt{\frac{\log k}{k}}. \quad (20)$$

Quicker asymptotic convergence to difference  $2(\omega^* - \omega) \rightarrow 1$  could be achieved in other ways, for example by the generalized Mermin-GHZ game [2] or by XORing together the answers (aka “XOR repetition”) of other pseudo-telepathy XOR games [5]. Although their bias scales in a weaker way, the APD games have the property that perfect entangled strategies exist for *any* choice of target bit strings  $s$ .

To investigate the incompleteness of the PREF condition, we define an XOR game that contains a PREF, but provably has commuting-operator value 1. This game is solved by a single-qubit strategy employing measurements in the  $X$ ,  $Y$ , and  $Z$  bases. This may be a starting point for stricter necessary criterion, building towards a complete algorithm for deciding the value of entangled XOR games. The existence of this game proves the following theorem.

► **Theorem 25.** *The PREF characterization is incomplete. In particular, there exists an XOR game with six players, alphabet size three, for which the entangled value is 1, but the noPREF condition is unable to detect this.*



Finally, we investigate thresholds in  $\omega^*$  by considering the behavior of randomly generated XOR games with a large number of clauses. We prove Theorem 26, which shows that (much like the classical case) random XOR games become unsatisfiable with high probability when  $m$  is larger than some constant times  $n$ . Previous techniques could show that  $k$ -XOR instances were unsatisfiable only in the “dense” regime, i.e. where  $m \geq \Omega(n^k)$  [24].

► **Theorem 26.** *For every  $k$ , there exists a constant  $C_k^{unsat}$  depending only on  $k$  such that a random  $k$ -XOR game  $G$  with  $m \geq C_k^{unsat}n$  clauses has value  $\omega^*(G) < 1$  with probability  $1 - o(1)$ .*

**Proof.** Explicit construction of a refutation presented in the full paper. ◀

We also investigate the average case performance of ncSoS. We show that random games with a fixed ratio of  $m$  to  $n$  have a minimal length refutation that scales like  $\Omega(n \log(n) / \log(\log(n)))$ , implying that it takes the ncSoS algorithm superexponential time to show that these games have  $\omega^* < 1$  (Theorem 27). These results should be thought of as quantum analogues of Grigoriev’s [17] integrality gap instances for classical XOR games.

► **Theorem 27.** *For any constant  $C$ , the minimum length refutation of a random 3-XOR game with  $m = Cn$  queries on an alphabet of size  $n$  has length at least*

$$\frac{en \log(n)}{8C^2 \log(\log(n))} - o\left(\frac{n \log(n)}{\log(\log(n))}\right) \tag{21}$$

*with probability  $1 - o(1)$  (as  $n \rightarrow \infty$ ). Hence, either  $\omega^* = 1$  or  $\Omega(n \log(n) / \log(\log(n)))$  levels of the ncSoS hierarchy are needed to witness that  $\omega^* < 1$  for such games.*

## 6 Future Work

We see four main directions in which our characterization of non-local XOR games could be extended.

First, our linear algebraic characterization of  $\omega^* = 1$  games is incomplete: there exist games with  $\omega^* = 1$  for which a MERP strategy cannot achieve value 1. We expect a strengthening of the PREF condition may allow us to extend our decidability algorithm to detect these games and develop dual strategies that solve them. Understanding the structure of such games would give further intuition about the behavior of optimal XOR commuting-operator strategies, in particular strategies which may require more entanglement than the simple MERP strategies.

Second, determining whether  $\omega^* = 1$  for nonsymmetric XOR games may be outside P or even undecidable. In the realm of Binary Constraint System (BCS) games, [27] shows that determining whether a general BCS game has perfect value is undecidable. The structural similarity between BCS games and XOR games suggests that perhaps some of the group theoretic techniques of that work could be applied to XOR games to arrive at a similar conclusion. An interesting class of games which may serve as an intermediate class between XOR and BCS games are “incomplete” XOR games in which there are  $k$  players but queries can involve  $< k$  variables, effectively ignoring some players. Even for  $k = 2$ , Tsirelson’s semidefinite programming characterization of  $\omega^*$  does not apply to incomplete XOR games, although in this case it is still easy to decide whether  $\omega^* = 1$ .

Third, while in this work we have focused on computing the entangled game value  $\omega^*$ , our methods may also be useful from the perspective of Bell inequalities, in which the quantity of interest is the maximal violation achievable by an entangled strategy. While this has



conventionally been measured in terms of the bias ratio  $(\omega^* - 1/2)/(\omega - 1/2)$ , the difference  $2(\omega^* - \omega)$  is an equally natural measure, and we hope that our techniques will render it more amenable to analysis. Indeed, in addition to the construction of Asymptotically Perfect Difference games mentioned above, our results have the following simple consequence: for symmetric games with  $\omega^* = 1$ , our characterization of the optimal strategies (MERP) together with the Grothendieck-type inequality of [3] imply that the bias ratio and difference are both bounded by constants depending only on  $k$ , and that for the difference, this constant is strictly less than one.

Finally, our results are almost entirely concerned with the question of determining whether  $\omega^* = 1$  or  $\omega^* < 1$ . However, we note that the MERP family of strategies includes the optimal strategy for the CHSH game [7] and more generally any multiplayer game with question and answer alphabet size two [31], but not for all XOR games [23]. It is an interesting open question to fully characterize when MERP strategies are optimal. In this setting there are still many classical tools which we do not know how to extend to the quantum case. As an example, consider *overconstrained* games in which there are many more constraints than variables and the signs of those constraints are chosen randomly. In the classical case, second moment methods can show that the value is close to  $1/2$  while in the quantum case we can only conclude that it is  $< 1$ .

---

## References

- 1 Jonathan Barrett, Lucien Hardy, and Adrian Kent. No signaling and quantum key distribution. *Physical review letters*, 95(1):010503, 2005. [arXiv:quant-ph/0405101](#).
- 2 Gilles Brassard, Anne Broadbent, and Alain Tapp. Recasting mermin’s multi-player game into the framework of pseudo-telepathy. *ArXiv e-prints*, 2004. [arXiv:quant-ph/0408052](#).
- 3 Jop Briët, Harry Buhrman, Troy Lee, and Thomas Vidick. Multipartite Entanglement in XOR Games. *Quantum Info. Comput.*, 13(3-4):334–360, March 2013. [arXiv:0911.4007](#).
- 4 Jop Briët and Thomas Vidick. Explicit lower and upper bounds on the entangled value of multiplayer XOR games. *Communications in Mathematical Physics*, 321(1):181–207, 2013. [arXiv:1108.5647](#).
- 5 Anne Lise Broadbent. Quantum pseudo-telepathy games. Master’s thesis, Université de Montréal, 2004.
- 6 Boris S Cirel’son. Quantum generalizations of Bell’s inequality. *Letters in Mathematical Physics*, 4(2):93–100, 1980.
- 7 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](#).
- 8 Richard Cleve, Peter Hoyer, Benjamin Toner, and John Watrous. Consequences and Limits of Nonlocal Strategies. In *CCC ’04*, pages 236–249, 2004. [arXiv:quant-ph/0404076](#).
- 9 Richard Cleve and Rajat Mittal. Characterization of binary constraint system games. In *International Colloquium on Automata, Languages, and Programming*, pages 320–331. Springer, 2014.
- 10 Roger Colbeck. *Quantum And Relativistic Protocols For Secure Multi-Party Computation*. PhD thesis, University of Cambridge, 2006. [arXiv:0911.3814](#).
- 11 Andrew C. Doherty, Yeong-Cherng Liang, Ben Toner, and Stephanie Wehner. The Quantum Moment Problem and Bounds on Entangled Multi-prover Games. In *CCC ’08*, pages 199–210, 2008. [arXiv:0803.4373](#).
- 12 Olivier Dubois and Jacques Mandler. The 3-XORSAT threshold. *Comptes Rendus Mathématique*, 335(11):963–966, 2002.

- 13 Artur K Ekert. Quantum cryptography based on Bell's theorem. *Physical review letters*, 67(6):661, 1991.
- 14 Joseph Fitzsimons, Zhengfeng Ji, Thomas Vidick, and Henry Yuen. Quantum proof systems for iterated exponential time, and beyond. *arXiv*, 2018. [arXiv:1805.12166](#).
- 15 Tobias Fritz, Tim Netzer, and Andreas Thom. Can you compute the operator norm? *Proceedings of the American Mathematical Society*, 142(12):4265–4276, 2014. [arXiv:1207.0975](#).
- 16 Daniel M Greenberger, Michael A Horne, Abner Shimony, and Anton Zeilinger. Bell's theorem without inequalities. *American Journal of Physics*, 58(12):1131–1143, 1990.
- 17 Dima Grigoriev. Linear lower bound on degrees of Positivstellensatz calculus proofs for the parity. *Theoretical Computer Science*, 259:613–622, 2001. [doi:10.1016/S0304-3975\(00\)00157-2](#).
- 18 Johan Håstad. Some optimal inapproximability results. *Journal of the ACM (JACM)*, 48(4):798–859, 2001.
- 19 Zhengfeng Ji. Classical verification of quantum proofs. In *Proceedings of the 48th Annual ACM SIGACT Symposium on Theory of Computing*, pages 885–898. ACM, 2016.
- 20 Zhengfeng Ji. Compression of Quantum Multi-Prover Interactive Proofs. *arXiv*, 2016. [arXiv:1610.03133](#).
- 21 Anand Natarajan and Thomas Vidick. Low-degree testing for quantum states, and a quantum entangled games PCP for QMA. *arXiv*, 2018. [arXiv:1801.03821v2](#).
- 22 Miguel Navascués, Stefano Pironio, and Antonio Acín. A convergent hierarchy of semidefinite programs characterizing the set of quantum correlations. *New J. Phys.*, 10(7):073013, 2008. [arXiv:0803.4290](#).
- 23 Dimiter Ostrev and Thomas Vidick. Entanglement of approximate quantum strategies in XOR games, 2016. [arXiv:1609.01652](#).
- 24 C. Palazuelos and T. Vidick. Survey on nonlocal games and operator space theory. *Journal of Mathematical Physics*, 57(1):015220, January 2016. [doi:10.1063/1.4938052](#).
- 25 David Pérez-García, Michael M Wolf, Carlos Palazuelos, Ignacio Villanueva, and Marius Junge. Unbounded violation of tripartite Bell inequalities. *Communications in Mathematical Physics*, 279(2):455–486, 2008. [arXiv:quant-ph/0702189](#).
- 26 Ben W Reichardt, Falk Unger, and Umesh Vazirani. Classical command of quantum systems. *Nature*, 496(7446):456–460, 2013.
- 27 William Slofstra. Tsirelson's problem and an embedding theorem for groups arising from non-local games, 2016. [arXiv:1606.03140](#).
- 28 Boris S Tsirel'son. Quantum analogues of the Bell inequalities. The case of two spatially separated domains. *Journal of Mathematical Sciences*, 36(4):557–570, 1987.
- 29 Umesh Vazirani and Thomas Vidick. Fully device-independent quantum key distribution. *Physical review letters*, 113(14):140501, 2014.
- 30 Thomas Vidick. Three-Player Entangled XOR Games Are NP-Hard to Approximate. In *Proceedings of the 2013 IEEE 54th Annual Symposium on Foundations of Computer Science, FOCS '13*, pages 766–775. IEEE Computer Society, 2013. [doi:10.1109/FOCS.2013.87](#).
- 31 Reinhard F Werner and Michael M Wolf. All-multipartite Bell-correlation inequalities for two dichotomic observables per site. *Physical Review A*, 64(3):032112, 2001.

# Testing Local Properties of Arrays

Omri Ben-Eliezer

Tel Aviv University, Tel Aviv 69978, Israel  
omrib@mail.tau.ac.il

---

## Abstract

We study testing of local properties in one-dimensional and multi-dimensional arrays. A property of  $d$ -dimensional arrays  $f : [n]^d \rightarrow \Sigma$  is  $k$ -local if it can be defined by a family of  $k \times \dots \times k$  forbidden consecutive patterns. This definition captures numerous interesting properties. For example, monotonicity, Lipschitz continuity and submodularity are 2-local; convexity is (usually) 3-local; and many typical problems in computational biology and computer vision involve  $o(n)$ -local properties.

In this work, we present a generic approach to test all local properties of arrays over any finite (and not necessarily bounded size) alphabet. We show that any  $k$ -local property of  $d$ -dimensional arrays is testable by a simple canonical one-sided error non-adaptive  $\epsilon$ -test, whose query complexity is  $O(\epsilon^{-1} k \log \frac{en}{k})$  for  $d = 1$  and  $O(c_d \epsilon^{-1/d} k \cdot n^{d-1})$  for  $d > 1$ . The queries made by the canonical test constitute sphere-like structures of varying sizes, and are completely independent of the property and the alphabet  $\Sigma$ . The query complexity is optimal for a wide range of parameters: For  $d = 1$ , this matches the query complexity of many previously investigated local properties, while for  $d > 1$  we design and analyze new constructions of  $k$ -local properties whose one-sided non-adaptive query complexity matches our upper bounds. For some previously studied properties, our method provides the first known sublinear upper bound on the query complexity.

**2012 ACM Subject Classification** Theory of computation  $\rightarrow$  Sketching and sampling

**Keywords and phrases** Property Testing, Local Properties, Monotonicity Testing, Hypergrid, Pattern Matching

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2019.11

**Related Version** A full version of the paper is available at [5], <https://eccc.weizmann.ac.il/report/2018/196/>.

**Acknowledgements** I would like to thank Frederik Benzing, Eric Blais, Eldar Fischer, Sofya Raskhodnikova, Daniel Reichman and C. Seshadhri for stimulating discussions, and the anonymous reviewers for helpful suggestions.

## 1 Introduction

Property testing [22, 33] is devoted to understanding how much information one needs to extract from an object in order to determine whether it satisfies a given property or is far from satisfying the property. This active research area has seen many developments through the last two decades; see the recent book of Goldreich [21] for a good introduction. The property testing notation we use here is standard, see Subsection 1.7 for the relevant definitions.

In this paper we focus on testing of local properties in structured data. The objects we consider are  $d$ -dimensional arrays, where  $d$  is a positive integer, viewed as a constant. A  $d$ -dimensional array of width  $n$ , or an  $[n]^d$ -array in short, is a function  $A : [n]^d \rightarrow \Sigma$  from the hypergrid  $[n]^d$  to the alphabet  $\Sigma$ , where the alphabet  $\Sigma$  is allowed to be any (arbitrarily large)



© Omri Ben-Eliezer;  
licensed under Creative Commons License CC-BY  
10th Innovations in Theoretical Computer Science (ITCS 2019).  
Editor: Avrim Blum; Article No. 11; pp. 11:1–11:20



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

finite set; we stress that the size of  $\Sigma$  is usually *not required* to be bounded as a function of the other parameters. For example, a *string* is an  $[n]^1$ -array, and the commonly used RGB representation of images is basically an  $[n]^2$ -array over  $\{0, 1, \dots, 255\}^3$ , where the three values corresponding to each pixel represent the intensity of red, green and blue in it.

## 1.1 Local properties

Korman, Reichman, and the author [7] recently investigated testing of the property of *consecutive pattern freeness*, i.e., not containing a copy of some (predefined) “forbidden” consecutive subarray. Here, a  $[k]^d$ -array  $S$  is a (consecutive) *subarray* of an  $[n]^d$ -array  $A$  in *location*  $(i_1, \dots, i_d) \in [n - k + 1]^d$  if  $A(i_1 + j_1 - 1, \dots, i_d + j_d - 1) = S(j_1, \dots, j_d)$  for any  $j_1, \dots, j_d \in [k]$ .

Naturally, a more general follow-up question raised in [7] was the following: what can be said about testing of properties defined by a *family of forbidden consecutive patterns*? As we shall see soon, many interesting properties of arrays (including a large fraction of the array properties that were previously investigated in the literature) can be characterized this way.

With this in mind, we call a property *local* if it can be characterized by a family of small forbidden consecutive patterns. Formally, a property  $\mathcal{P}$  of  $[n]^d$ -arrays over an alphabet  $\Sigma$  is *k-local* (for  $2 \leq k \leq n$ ) if there exists a family  $\mathcal{F}$  of  $[k]^d$ -arrays over  $\Sigma$  so that the following holds for any  $[n]^d$ -array  $A$  over  $\Sigma$ :

$$A \text{ satisfies } \mathcal{P} \iff \text{None of the (consecutive) subarrays of } A \text{ is in } \mathcal{F}.$$

For  $\mathcal{P}$  as above, we sometimes write  $\mathcal{P} = \mathcal{P}(\mathcal{F})$  to denote that  $\mathcal{P}$  is defined by the forbidden family  $\mathcal{F}$ .

The main contribution of this work is a generic one-sided error non-adaptive framework to test *k-local* properties. In some cases, our method either matches or beats the best known upper bounds on the query complexity (although the running time might be far from optimal in general). We show the optimality of our method by proving a matching lower bound for non-adaptive one-sided tests, as well as a (weaker) lower bound for two-sided tests.

In order to demonstrate the wide range of properties captured by the above definition, we now present various examples of properties that are *k-local* for small *k*, including some of the most widely investigated properties in the property testing literature, as well as properties from areas of computer science that were not systematically studied in the context of property testing. In what follows, the sum of two tuples  $x = (x_1, \dots, x_d), y = (y_1, \dots, y_d)$  is defined as the tuple  $(x_1 + y_1, \dots, x_d + y_d)$ ; additionally,  $e^i$  denotes the *i*-th unit vector in *d* dimensions.

**Monotonicity.** Perhaps the most thoroughly investigated property in the testing literature: see e.g. the entries related to monotonicity testing in the Encyclopedia of Algorithms [14, 32] and the references within. An  $[n]^d$ -array  $A$  over an ordered alphabet  $\Sigma$  is *monotone* (non-decreasing) if  $A(x) \leq A(y)$  for any  $x = (x_1, \dots, x_d)$  and  $y = (y_1, \dots, y_d)$  satisfying  $x_i \leq y_i$  for any *i*. Monotonicity is 2-local: an array  $A$  is monotone if and only if there is no pair  $x, x + e^i \in [n]^d$  so that  $A(x) > A(x + e^i)$ .

**Lipschitz continuity.** Another well-investigated property with connections to differential privacy [3, 10, 15, 24], an  $[n]^d$ -array  $A$  is *c-Lipschitz continuous* if  $|A(x) - A(y)| \leq c \sum_{i=1}^d |y_i - x_i|$  for any  $x, y \in [n]^d$ . This condition holds iff  $|A(x) - A(x + e^i)| \leq c$  for any  $x, x + e^i \in [n]^d$ , and thus Lipschitz continuity is also 2-local.

**Convexity.** Discrete convexity is an important geometric property with connections to optimization and other areas [9, 8, 13, 18, 30, 31]. A one-dimensional array  $A$  is *convex* if  $\lambda A(x) + (1 - \lambda)A(y) \geq A(\lambda x + (1 - \lambda)y)$  for any  $x, y \in [n]$  and  $0 < \lambda < 1$  satisfying

$\lambda x + (1 - \lambda)y \in [n]$ . Convexity is 3-local for the case  $d = 1$ : an array  $A : [n] \rightarrow \Sigma$  is convex if and only if  $A[x] - 2A[x + 1] + A[x + 2] \geq 0$  for any  $x \in [n - 2]$ . In higher dimensions, several different notions of discrete convexity have been used in the literature – see e.g. the introductory sections of the book of Murota on discrete convex analysis [27]. Two of the commonly used definitions,  $M^\#$ -convexity and  $L^\#$ -convexity, are 3-local and 4-local, respectively: see Theorems 4.1 and 4.2 in [26], where it is shown that both notions can be defined locally using slight variants of the Hessian matrix consisting of the partial discrete derivatives. Another common definition that is a natural variant of the continuous case states that convexity is equivalent to the positive semi-definiteness of the Hessian matrix; under this definition, convexity is 3-local. A strictly weaker notion of convexity, called *separate convexity* [13], is defined as follows: an  $[n]^d$ -array  $A$  is separately convex if it is convex along each of the axes. Similarly to one-dimensional convexity, separate convexity is 3-local for any  $d$ .

**Properties of higher order derivatives.** More generally, *any* property of arrays that can be characterized by “forbidden pointwise behavior” of the first  $k$  discrete derivatives [13] is  $(k + 1)$ -local. Monotonicity (for  $k = 1$ ), Lipschitz continuity ( $k = 1$ ) and convexity ( $k = 2$ ) are special cases of such properties.

**Submodularity.** An important property closely related to convexity [10, 11, 30, 34]. Given  $x = (x_1, \dots, x_d), y = (y_1, \dots, y_d) \in [n]^d$ , define  $x \wedge y = (\min(x_1, y_1), \dots, \min(x_d, y_d))$  and  $x \vee y = (\max(x_1, y_1), \dots, \max(x_d, y_d))$ . An  $[n]^d$ -array is *submodular* if  $A[x \wedge y] + A[x \vee y] \leq A[x] + A[y]$  for any  $x, y \in [n]^d$ . Submodularity is 2-local: it is not hard to verify that submodularity is equivalent to the condition that  $A(x) + A(x + e^i + e^j) \leq A(x + e^i) + A(x + e^j)$  for all  $x$ .

**Pattern matching and computer vision.** Tasks involving pattern matching under some limitations – such as noise in the image, obstructed view, or rotation of elements in the image – are at the core of computer vision and its applications. For example, the local property of not containing a good enough  $\ell_1$ -approximation of a given forbidden pattern is of practical importance in computer vision. Sublinear approaches closely related to property testing are known to be effective for problems of this type, see e.g. [25].

**Computational biology.** Many problems in computational biology are closely related to one-dimensional pattern matching. As an example, a defensive mechanism of the human body against RNA-based viruses involves “cutting” a suspicious RNA fragment, if it finds one of a (small) family of short forbidden consecutive patterns in it, indicating that this RNA might belong to a virus. Thus, in order to generate fragments of RNA that are not destroyed by such defensive mechanisms (which is a basic task in computational biology), understanding the process of “repairing” a fragment so that it will not contain any of the forbidden patterns is an interesting problem related to property testing.

## 1.2 Previous results on local properties

### One-dimensional arrays

A seminal result of Ergün et al. [19] shows that for constant  $\epsilon$ , monotonicity is  $\epsilon$ -testable over the line (that is, for one-dimensional arrays) using  $O(\log n)$  queries over general alphabets. The non-adaptive one-sided error test proposed in [19] is based, roughly speaking, on imitating a binary search non-adaptively. It was shown by Fischer [20] that the above is tight even for two-sided error adaptive tests, proving a matching  $\Omega(\log n)$  lower bound. Later on, Parnas, Ron and Rubinfeld [30] and Jha and Raskhodnikova [24] showed that the  $O(\log n)$  upper bound on the non-adaptive one-sided query complexity also holds for convexity and Lipschitz

## 11:4 Testing Local Properties of Arrays

continuity, respectively. For general  $\epsilon$ , the upper bound in [24] is of the type  $O(\epsilon^{-1} \log n)$ ; the same work also presents a matching lower bound of  $\Omega(\log n)$  for the one-sided non-adaptive case, while  $\Omega(\log n)$  lower bounds for two-sided non-adaptive tests of convexity, and more generally, monotonicity of the  $\ell$ -th derivative, are proved by Blais, Raskhodnikova and Yaroslavtsev [13] using a communication complexity based approach [12]. Finally, a recent result of Belovs [4] refines the one-sided non-adaptive query complexity of monotonicity to  $O(\epsilon^{-1} \log \epsilon n)$ .

When the alphabet is binary (of size two), general positive results are known regarding the testability of local properties in one dimension. It follows from the testability of regular languages, established by Alon et al. [2], that any  $k$ -local property is testable in  $O(c(\mathcal{F})\epsilon^{-1}(\log^3(\epsilon^{-1})))$  queries, where  $c(\mathcal{F})$  depends only on the family  $\mathcal{F}$  of forbidden consecutive length- $k$  patterns defining the property. However,  $c(\mathcal{F})$  can be exponential in  $k$  in general.

### Multi-dimensional arrays

Chakrabarty and Seshadhri [16] extended some of the above results to hypergrids, showing that a general class of so-called “bounded derivative” properties (all of which are 2-local), including monotonicity and Lipschitz continuity as special cases, are all testable over  $[n]^d$ -arrays with  $O(\epsilon^{-1} d \log n)$  queries. Another work by the same authors [17] shows a matching lower bound of  $\Omega(\epsilon^{-1} d \log \epsilon n)$  for monotonicity, that holds even for two-sided adaptive tests, while the communication complexity approach of [13] gives a (non-adaptive, two-sided)  $\Omega(d \log n)$  lower bound for convexity, separate convexity and Lipschitz.

Submodularity is testable for  $d = 2$  with  $O(\log^2 n)$  queries [30]; However, no non-trivial upper bound on the query complexity is known for submodularity in the case  $d > 2$  and convexity in the case  $d > 1$  under the Hamming distance and over general alphabets (although [9] proves constant-query testability for 2D convexity over a binary alphabet). Under  $L_1$ -distance and for any  $d$ , it was shown in [10] that convexity in  $[n]^d$ -arrays is testable with number of queries that depends only on  $d$ .

### Pattern freeness

In [7], it was shown that the property of (consecutive) pattern freeness, for a single forbidden pattern, is testable with  $O(c_d/\epsilon)$  queries for any  $d$ . The proof, however, requires multiple sophisticated combinatorial observations and does not seem to translate to the case of a family of forbidden patterns discussed here.

## 1.3 Our results

In this work, we present a generic approach to test *all*  $k$ -local properties of  $[n]^d$ -arrays. Among other consequences, a simple special case of our result in the one-dimensional regime shows that the abundance of properties whose query complexity is  $\Theta(\log n)$  is not a coincidence: in fact, *any*  $O(1)$ -local property of one-dimensional arrays is testable with  $O(\log n)$  queries, using a canonical binary search like querying scheme.

On the other hand, we prove a lower bound for testing local properties in  $d > 1$  dimensions, showing that the query complexity of our test is optimal (for fixed  $d$ ) among non-adaptive one-sided tests, even when restricted to alphabets of size that is polynomial in  $n^d$ . We also prove a lower bound for non-adaptive two-sided tests.

### 1.3.1 Upper bounds

Our first main result is an upper bound on the number of queries required to test any  $k$ -local property of  $[n]^d$ -arrays non-adaptively with one-sided error. The test is *canonical* in a strong sense: The queries it makes depend on  $n, d, k$ , and (relatively weakly) on  $\epsilon$ ; they do not depend on  $\mathcal{P}$  or the alphabet  $\Sigma$ . In other words, it makes the same type of queries for all  $k$ -local properties of  $[n]^d$ -arrays over any finite (and not necessarily bounded-size) alphabet.

► **Theorem 1.** *Let  $2 \leq k \leq n$  and  $d \geq 1$  be integers, and let  $\epsilon > 0$ . Any  $k$ -local property  $\mathcal{P}$  of  $[n]^d$ -arrays over any finite (and not necessarily bounded size) alphabet has a one-sided error non-adaptive  $\epsilon$ -test whose number of queries is*

- $O\left(\frac{k}{\epsilon} \cdot \log \frac{\epsilon n}{k}\right)$  for  $d = 1$ .
- $O\left(c^d \frac{k}{\epsilon^{1/d}} \cdot n^{d-1}\right)$  for  $d > 1$ .

Here,  $c > 0$  is an absolute constant. The test chooses which queries to make based only on the values of  $n, d, k, \epsilon$ , and independently of the property  $\mathcal{P}$  and the alphabet  $\Sigma$ .

Note that we are interested here in the domain where  $n$  is large and  $d$  is considered a constant. Thus, we did not try to optimize the  $c^d$  term in the second bullet, seeing that it is negligible compared to  $n^{d-1}$  anyway.

#### Running time

The main drawback of our approach is the running time of the test, which is high in general. After making all of its queries, our test runs an *inference* step, where it tries to evaluate (by enumerating over all relevant possibilities) whether a violation of the property must occur in view of the queries made, and reject if this is the case.

Without applying any property-specific considerations, the running time of the inference step is of order  $|\Sigma|^{O(n^d)}$ . However, for various specific properties of interest, such as monotonicity and 1D-convexity, it is not hard to make the running time of the inference step of the same order of magnitude as the query complexity. Moreover, in one dimension we can use dynamic programming to achieve running time that is significantly better than the naive one, but still much higher than the query complexity in general:  $O(|\Sigma|^{O(k)}n)$ . This works for any  $k$ -local property in one dimension; see the last part of Subsection 1.4.1 for more details.

#### Proximity oblivious test

Interestingly, the behavior of the test depends quite minimally on  $\epsilon$ , and it can be modified very slightly to create a proximity oblivious test (POT) for any  $k$ -local property. The useful notion of a POT, originally defined by Goldreich and Ron [23], refers to a test that does *not* receive  $\epsilon$  as an input, and whose success probability for an input not satisfying the property is a function of the Hamming distance of the input from the property.

► **Theorem 2.** *Fix  $d > 0$ . Any  $k$ -local property  $\mathcal{P}$  of  $[n]^d$ -arrays over any finite (but not necessarily bounded size) alphabet has a one-sided error non-adaptive proximity oblivious test whose number of queries is  $O(k \log(n/k))$  if  $d = 1$  and  $O(kn^{d-1})$  if  $d > 1$ . For any input  $A$  not satisfying  $\mathcal{P}$ , the rejection probability of  $A$  is linear (for fixed  $d$ ) in the Hamming distance of  $A$  from  $\mathcal{P}$ .*

One can run  $O(c_d/\epsilon)$  iterations of the POT to obtain a standard one-sided error non-adaptive test. The query complexity is  $O(k\epsilon^{-1} \log(n/k))$  for  $d = 1$  and  $O(c_d k \epsilon^{-1} n^{d-1})$  for  $d > 1$ , where  $c_d > 0$  depends only on  $d$ . Thus, the POT-based test is sometimes as good as the test of Theorem 1 (specifically, for  $d = 1$  it matches the above bounds for almost the whole range of  $\epsilon$  and  $k$ ). In any case, the multiplicative overhead of the POT-based test is sublinear in  $1/\epsilon$  across the whole range.

### Type of queries

In one dimension, many of the previously discussed properties, including, for example, monotonicity and Lipschitz continuity, are testable in  $O(\log n)$  queries. Previously known tests for monotonicity and Lipschitz continuity make queries that resemble a binary search: these tests query pairs of entries of distance  $2^i$  for multiple choices of  $0 \leq i \leq \log n$ .

Our test continues the line of works using querying schemes roughly inspired by binary search. The test queries structures that can be viewed, intuitively, as  $L_\infty$ -spheres of different sizes in  $[n]^d$ . For this purpose, an  $L_\infty$ -sphere with radius  $r$  and width  $\ell$  in  $[n]^d$  is a set  $X_1 \times X_2 \times \dots \times X_d \subseteq [n]^d$ , where each  $X_i$  is a union of intervals of the form  $[a_i, a_i + 1, \dots, a_i + \ell - 2, a_i + \ell - 1] \cup [b_i - \ell + 2, b_i - \ell + 3, \dots, b_i - 1, b_i]$ , and  $b_i - a_i \in \{r, r + 1\}$  for any  $i \in [d]$ . More specifically, our test for  $k$ -local properties queries spheres with width  $k - 1$  and radius of order  $2^i$  for different values of  $i$ . In the simple special case where  $d = 1$  and  $k = 2$ , this is very similar to the querying scheme mentioned in the previous paragraph.

### Implications

In one dimension, the query complexity of the test matches the best known upper bounds (and, in some regimes, refines the dependence on  $\epsilon$ ) for several previously investigated properties including monotonicity, Lipschitz continuity and convexity. For monotonicity of  $k$ -th order derivatives, which is  $(k + 1)$ -local, it proves the first sublinear upper bound on the query complexity:  $O(k \log n)$ ; in comparison, the best known lower bound [13] is  $\Omega(\log n)$ .

For pattern matching type properties in 1D arrays (including applications in computational biology and other areas), our approach gives a property- and alphabet-independent upper bound of  $O(k \log n)$  on the query complexity, with essentially optimal dependence on  $\epsilon$  as well. Previously known approaches for testing such properties, like the regular languages testing approach [2], yield tests whose query complexity is dependent on the family of forbidden patterns considered, whose size might be exponential in the locality parameter  $k$ . Our approach, on the other hand, requires an  $O(\log n)$  “overhead”, but its query complexity is independent of the size of the forbidden family discussed. Instead, the dependence in  $k$  is linear.

In multiple dimensions, our approach is far from tight for well-understood properties such as monotonicity and Lipschitz continuity, whose query complexity is known to be  $\Theta(d \log n)$  (in comparison, our approach yields an  $O(n^{d-1})$  type bound). However, for testing of other properties like convexity (for  $d > 1$ ) and submodularity (for  $d > 2$ ) in  $[n]^d$ -arrays, no non-trivial upper bounds on the query complexity are known over general alphabets, so our upper bound of  $O(n^{d-1})$  is the first such bound. While we do not believe this bound is tight in general, this might be a first step towards the development of new tools for efficiently testing such properties.

### Sketching for testing

The fact that the queries made are completely independent of the property suggests the following sketching technique allowing for “testing in retrospect”: Given  $\epsilon$  and  $k$  in advance, we make all queries of the generic  $\epsilon$ -test for  $k$ -local properties in “real time”, and store them for postprocessing. This is suitable, for example, in cases where we have limited access to a large input for a limited amount of time (e.g. when reading the input requires specialized expensive machinery), but the postprocessing time is not an issue. Note that for this approach we do not need to know the property of interest in advance.



### 1.3.2 Lower bounds

Our next main result is a tight lower bound for non-adaptive one-sided error testing of local properties, that applies for any  $d$ , and is tight for any fixed  $d$  satisfying  $d > 1$ .

► **Theorem 3 (One-sided tests).** *Let  $d \geq 1$  and  $n \geq k \geq 2$  be integers, and let  $d/n < \epsilon < 1$ . There exists a  $k$ -local property  $\mathcal{P}$  of  $[n]^d$ -arrays over an alphabet  $\Sigma$  of size  $n^{O(d)}$ , so that any non-adaptive one-sided error  $\epsilon$ -test for  $\mathcal{P}$  requires  $\Omega\left(\min\left\{\frac{k}{d\epsilon^{1/d}} \cdot n^{d-1}, n^d\right\}\right)$  queries.*

Note that the size of the alphabet in Theorem 3 is only polynomial in the input size. We also prove a lower bound for non-adaptive two-sided tests; here the dependence in  $|\Sigma|$  is exponential.

► **Theorem 4 (Two-sided tests).** *Let  $d \geq 1$  and  $n \geq k \geq 2$  be integers, and let  $d/n < \epsilon < 1$ . There exists a  $k$ -local property  $\mathcal{P}$  of  $[n]^d$ -arrays over an alphabet  $\Sigma$  of size  $2^{O(n^d)}$ , so that any non-adaptive two-sided error  $\epsilon$ -test for  $\mathcal{P}$  requires  $\Omega\left(\min\left\{\frac{\sqrt{k}}{d\epsilon^{(d+1)/2d}} \cdot n^{(d-1)/2}, n^d\right\}\right)$  queries.*

For fixed  $d > 1$ , the lower bound for one-sided tests matches the upper bound from Theorem 1 across the whole range of  $\epsilon$  and  $k$ . For  $d = 1$  the bound obtained here is  $\Omega(k/\epsilon)$ , which is tight up to a  $\log n$  factor. Note the threshold behavior occurring at  $k/\epsilon^{1/d} = \Theta(n)$ : When  $k/\epsilon^{1/d} = o(n)$ , the upper bound of Theorem 1 implies that any  $k$ -local property is  $\epsilon$ -testable with a sublinear number of queries, while for  $k/\epsilon^{1/d} = \Omega(n)$ , the property of Theorem 3 requires  $\Omega(n^d)$  non-adaptive one-sided queries to test.

From Theorem 4 we conclude that the improvement in query complexity obtained by two-sided error non-adaptive tests is at most quadratic in the worst case; specifically, there exists a 2-local property requiring  $n^{\Omega(d-1)}$  queries to test by two-sided non-adaptive tests.

## 1.4 Proof ideas and techniques

Here we present the main ideas of our proofs in an informal way, starting with the upper bound. For simplicity, we stick to the one-dimensional case, and assume that  $\epsilon$  is fixed and  $k = o(n)$ .

### 1.4.1 Upper bound for 1D

Suppose that  $\mathcal{P} = \mathcal{P}(\mathcal{F})$  is a  $k$ -local property of  $[n]^1$ -arrays  $A$  over an alphabet  $\Sigma$ , defined by the forbidden family  $\mathcal{F}$ . Let  $S$  be a consecutive subarray of  $A$  of length at least  $2k - 2$ . The *boundary* of  $S$  consists of the first  $k - 1$  elements and the last  $k - 1$  elements of  $S$ , and all other elements of  $S$  are its *interior*. We call  $S$  *unrepairable* if one cannot make the array  $S$  satisfy the property  $\mathcal{P}$  without changing the value of at least one element in its boundary. Otherwise,  $S$  is *repairable*. Observe the following simple facts.

- It suffices to only query the boundary elements of  $S$  in order to determine whether  $S$  is unrepairable.
- If  $S$  is unrepairable, then  $A$  does not satisfy  $\mathcal{P}$ .
- If  $S$  is repairable, then we can delete all forbidden patterns from  $S$  by modifying only entries in its interior, without creating any new copies of forbidden patterns in  $A$ .

We call the process of understanding whether  $S$  is unrepairable using only its boundary elements *inference*. Note that the inference step does not make any additional queries.

### A simple sublinear test

A first attempt at a generic test for local properties is the following: we query  $\Theta(\sqrt{n})$  intervals in  $[n]$ , each containing exactly  $k - 1$  consecutive elements, including the intervals  $\{1, \dots, k - 1\}$  and  $\{n - k + 2, \dots, n\}$ , where the distance between each two neighboring intervals is  $\Theta(\sqrt{n})$ . A *block* is a subarray consisting of all elements in a pair of neighboring intervals and all elements between them. The crucial observation is that at least one of the following must be true, for any array  $A$  that is  $\epsilon$ -far from  $\mathcal{P}$  (recall that  $\epsilon$  is fixed).

- At least one of the blocks is unrepairable.
- At least  $\Omega(\sqrt{n})$  of the blocks do not satisfy  $\mathcal{P}$ .

Indeed, if the first condition does not hold, then one can make  $A$  satisfy  $\mathcal{P}$  by only changing elements in the interiors of blocks that do not satisfy  $\mathcal{P}$ . Seeing that  $A$  is  $\epsilon$ -far from  $\mathcal{P}$  and that we do not need to modify elements in the interiors of blocks that satisfy  $\mathcal{P}$ , this implies that at least  $\Omega(\sqrt{n})$  of the blocks do not satisfy  $\mathcal{P}$ .

Now we are ready to present the test: We query all  $O(k\sqrt{n})$  elements of all intervals, and additionally, all  $O(\sqrt{n})$  elements of  $O(1)$  blocks. Querying all elements of all intervals suffices to determine (with probability 1) whether one of the blocks is unrepairable. If  $A$  is  $\epsilon$ -far from  $\mathcal{P}$  and does not contain unrepairable blocks, querying  $O(1)$  full blocks will catch at least one block not satisfying  $\mathcal{P}$  with constant probability, as desired.

For more details, see Section 3 and the preliminary Section 2 that prepares the required infrastructure.

### The optimal test

Improving the query complexity requires us to construct a system of *grids* – which are merely subsets of  $[n]$  – inspired by the behavior of binary search. In comparison, the approach of the previous test is essentially to work with a single grid.

The first (and coarsest) grid contains only the first  $k - 1$  elements and the last  $k - 1$  elements of  $[n]$ . In other words, it is equal to  $\{1, \dots, k - 1, n - k + 2, \dots, n\}$ . The second grid refines the first grid – that is, it contains all elements of the first grid – and additionally, it contains  $k - 1$  consecutive elements whose center is  $n/2$  (whenever needed, rounding can be done rather arbitrarily). We continue with the construction of grids recursively: To construct grid number  $i + 1$ , we take grid number  $i$  and add  $k - 1$  elements in the middle of each block of grid  $i$  (blocks are defined as before). Note that the length of blocks is roughly halved with each iteration. We stop the recursive construction when the length of all of the blocks becomes no bigger than  $ck$ , where  $c \geq 2$  is an absolute constant.

For each block  $B$  in grid number  $i > 1$ , we define its *parent*, denoted  $\text{Par}(B)$ , as the unique block in interval  $i - 1$  containing it. A block  $B$  in the system of grid is *maximally unrepairable* if it is unrepairable, and all blocks (of all grids in the system) strictly containing it are repairable. It is not hard to see that different maximally unrepairable blocks have disjoint interiors.

The main observation now is that in order to make  $A$  satisfy  $\mathcal{P}$ , it suffices to only modify entries in the interiors of parents of maximally unrepairable blocks. If  $A$  is  $\epsilon$ -far from  $\mathcal{P}$ , then the total length of these parents must therefore be  $\Omega(n)$  (for constant  $\epsilon$ ). However, since the length of  $\text{Par}(B)$  is roughly twice the length of  $B$ , we conclude that the total length of all maximally unrepairable blocks is  $\Omega(n)$ .

With this in hand, it can be verified that the following test has constant success probability. For each grid in the system, we pick one block of the grid uniformly at random, and query all entries of its boundary. Additionally, for the finest grid (whose block length is  $O(k)$ ), we also query all interior elements of the picked block.

For more details, see Section 4 (which builds on the infrastructure of Section 2).

### Running time in 1D

We now show that the running time of the inference step for a block of length  $m$  is  $m|\Sigma|^{O(k)}$ . Summing over all block lengths, this would imply that the total running time of the test is  $n|\Sigma|^{O(k)}$ .

The proof uses dynamic programming. Let  $S$  be an array of length  $m$  over  $\Sigma$ , and assume that  $S(1), S(2), \dots, S(k-1)$  and  $S(m-k+2), \dots, S(m)$  are all known. For each “level” from 1 to  $m-k+1$ , we keep a boolean predicate for each of the  $|\Sigma|^k$  possible patterns of length  $k$  over  $\Sigma$ . These predicates are calculated as follows.

- In the first level, the predicate of  $\sigma = (\sigma_1, \dots, \sigma_k)$  evaluates to TRUE if  $S(1) = \sigma_1, \dots, S(k-1) = \sigma_{k-1}$ , and additionally,  $\sigma \notin \mathcal{F}$ , that is,  $\sigma$  is not a forbidden pattern. Otherwise, the predicate of  $\sigma$  is set to FALSE.
- For  $i = 2$  to  $m-k+1$ , the predicate of  $\sigma = (\sigma_1, \dots, \sigma_k)$  in level  $i$  evaluates to TRUE if and only if
  1.  $\sigma \notin \mathcal{F}$ .
  2. there exists  $\sigma' = (\sigma'_0, \sigma_1, \dots, \sigma_{k-1})$  that evaluates to TRUE in level  $i-1$ .
- Finally, the predicates in level  $m-k+1$  are modified as follows: for all  $\sigma = (\sigma_1, \dots, \sigma_k)$  so that  $\sigma_j \neq S(m-k+j)$  for some  $j \geq 2$ , we set the predicate of  $\sigma$  to FALSE.

It is not hard to see that  $S$  is unrepairable if and only if all predicates at level  $m-k+1$  are FALSE. The running time is  $O(m|\Sigma|^{cd})$  for a suitable constant  $c > 0$ .

### Generalization to higher dimensions

The generalization to higher dimensions is relatively straightforward; the main difference is that the boundary of blocks now is much larger: blocks of size  $m \times \dots \times m$  have boundary of size  $O(kdm^{d-1})$ . Thus, essentially the same proof as above (with suitable adaptations of the definitions) yields a test with query complexity  $O(kdn^{d-1})$  for constant  $\epsilon$ . For the running time, we can no longer use dynamic programming; using the naive approach of enumerating over all possible interior elements of a block, we get that the inference time for a block of size  $m \times \dots \times m$  is  $|\Sigma|^{O(m^d)}$ , making the total running time of the test  $|\Sigma|^{O(n^d)}$ .

#### 1.4.2 Lower bound

The property  $\mathcal{P}$  underlying our lower bound construction consists of  $[n]^d$ -arrays  $A$  over  $\Sigma$  satisfying all of the following properties. Here we provide a construction over alphabet size  $2^{O(n^d)}$ , but in Section 6 of the full version of this paper [5] we show how a simple modification of the property can be conducted in order to decrease the size of the required alphabet to  $n^{O(d)}$  for the proof of Theorem 3 (unfortunately, for the proof of Theorem 4 this modification does not work).

- The alphabet  $\Sigma$  is of the form  $[n]^d \times [n]^d \times 2^{[2n^{d-1}]}$ , where  $2^X$  is the *power set* of a set  $X$ . The value of  $A$  in entry  $x \in [n]^d$  is represented as a tuple  $A_1(x), A_2(x), A_3(x)$ . We view  $A_1(x), A_2(x)$  as *pointers* emanating from  $x$ .
- For every  $x \in [n]^d$  we require  $A_1(x) = x$ . That is,  $A_1$  points to the location of the element itself.
- There exists a special location  $\ell = (\ell_1, \dots, \ell_d) \in [n]^d$  so that all  $x \in [n]^d$  point to  $\ell$  with their second pointer, that is,  $A_2(x) = \ell$ . We call this location the *lower center of gravity*. We define an *upper center of gravity* as  $u = (\ell_1 + 1, \ell_2, \dots, \ell_d)$ .

## 11:10 Testing Local Properties of Arrays

- A *floor* entry  $x = (x_1, \dots, x_d) \in [n]^d$  satisfies  $x_1 = 1$  and a *ceiling* entry satisfies  $x_1 = n$ . For each such floor or ceiling entry  $x$ , we pick  $A_3(x)$  to be a singleton (i.e., a set with one element).
- For each floor element  $x$ , there exists a path  $\Gamma_x$  from  $x$  to the lower center of gravity  $\ell$ . Similarly, for each ceiling element  $y$ , there is a path  $\Gamma_y$  directed towards the upper center of gravity,  $u$ . In both cases, the path is of length  $O(nd)$ . The structure of the path depends only on its start and end points (so depends only on  $x$  and  $\ell$  in the first case, and  $y$  and  $u$  in the second case).
- The  $A_3$ -data “flows” to the center of gravity through paths. Formally, the  $A_3$ -set of each entry  $y$  that is not a floor or ceiling entry is required to be equal to the union  $\bigcup_{x: y \in \Gamma_x} A_3(x)$ . In other words, the data in each location in  $[n]^d$  is an “aggregation” of the data flowing in all paths that intersect it.
- Finally, we require that  $A_3(u) = A_3(\ell)$ .

While  $\mathcal{P}$  was defined above in global terms, we show that it is actually a 2-local property, that is, all conditions specified here can be written in a 2-local way.

To prove the lower bound, we follow Yao’s minimax principle [35], defining a distribution of arrays satisfying  $\mathcal{P}$ , and a distribution of arrays which are  $\Omega(1)$ -far from satisfying  $\mathcal{P}$ , so that a large number of queries is required to distinguish between the distributions.

As positive examples, we take a collection of arrays  $A$  satisfying the property, and require that all the singletons in the floor are pairwise disjoint. For negative examples (that are  $1/4$ -far from  $\mathcal{P}$ ), we consider a collection of arrays satisfying all of the above requirements other than the last. Instead, all singletons in the floor and the ceiling are pairwise disjoint (so in particular,  $A_3(\ell) \cap A_3(u) = \emptyset$ ).

We show that for any given  $x \in [n]^d$ , the expected size of  $A_3(x)$  over each of the distributions is  $O(d)$ . For the one-sided error case, it is shown that one needs to know the values of at least  $\Omega(n^{d-1})$  singletons to be able to distinguish between positive and negative examples with one sided error, implying that  $\Omega(n^{d-1}/d)$  queries are required to reject negative examples with constant probability.

For the two sided error case, the argument is inspired by the birthday paradox. Very loosely speaking, it follows from the fact that, given two unknown unordered sets  $A$  and  $B$  of size  $n$ , one has to make  $\Omega(\sqrt{n})$  queries to distinguish between the case that  $A = B$  and the case that  $A \cap B = \emptyset$ .

Due to space considerations, the full proofs of Theorems 3 and 4 are relegated to the full version of this paper [5]; see Sections 5 and 6 there.

### 1.5 Other related work

This subsection complements Subsection 1.2, presenting other related previous works that were not mentioned above.

#### General results in property testing

This paper adds to the growing list of general characterization results in property testing of strings, images, and multi-dimensional arrays; see [1, 6] and the references within for characterization-type results in these domains, mostly over a fixed size alphabet. In particular, for strings, it was shown by Alon et al. [2] that any local property over a fixed size alphabet is constant-query testable, and this paper shows that an overhead of at most  $O(\log n)$  is required when the alphabet size is unbounded.

## Hyperfiniteness

A graph is hyperfinite if, roughly speaking, it can be decomposed into constant size connected components by deleting only a small constant fraction of the edges. Newman and Sohler [28] investigated the problem of testing in hyperfinite graphs, showing that any property of hyperfinite bounded degree graphs is testable with a constant number of queries. While the graph with which we (implicitly) work – the hypergrid graph, whose vertices are in  $[n]^d$  and two vertices are neighbors if they differ by 1 in one coordinate – is a hyperfinite bounded degree graph (for constant  $d$ ), the results of [28] are incomparable to ours. Indeed, in our case the vertices are inherently ordered, and it does not make sense to allow adding edges between vertices that are not neighbors (as entries of  $[n]^d$ ), unlike the case in [28], where one may add or remove edges arbitrarily between any two vertices. Still, the hyperfiniteness of our graph seems to serve as a major reason that local properties have sublinear tests.

## Block tests for image properties

The works of Berman, Murzabulatov and Raskhodnikova [9] and Korman, Reichman, and the author [7] on testing of image properties (that is, on visual properties of 2D arrays) show that tests based on querying large consecutive blocks are useful for image property testing. In this work, the general queries we make are quite different: we query the *boundaries* of blocks of different sizes, so the queries are *spherical*, in the sense that a block can be seen as a ball in the  $L_\infty$ -metric on vectors in  $[n]^d$ , while its boundary can be seen as the (width- $k$ ) sphere surrounding this ball. This introduces a new type of queries shown to be useful for image property testing.

## 1.6 Discussion

### Small alphabets

The results in this work are alphabet independent, and in particular, they work for alphabets over any size. An intriguing direction of research is to understand whether one can obtain more efficient general testability results for local properties of multi-dimensional arrays over smaller alphabets; this line of research has been conducted for specific properties of interest, like monotonicity and convexity [4, 29]. Note that the one-sided non-adaptive lower bound we prove here can be adapted to yield a  $|\Sigma|^{\Omega(1)}$  lower bound for testing local properties over alphabets  $\Sigma$  of size smaller than  $n^d$ .

The most interesting special case is that of constant-sized (and in particular, binary) alphabets. Here, no lower bounds that depend on  $n$  are known. For the case  $d = 1$ , it is known that all  $O(1)$ -local properties are constant query testable; this follows from a result of Alon et al. [2], who showed that any regular language is constant-query testable. However, it is not known whether an analogous statement holds in higher dimensions. That is, for any  $d > 1$ , the question whether all  $k$ -local properties of  $[n]^d$ -arrays over  $\{0, 1\}$  are  $\epsilon$ -testable with query complexity that depends only on  $d$ ,  $k$ , and  $\epsilon$ , first raised in [7] (see also [1]), remains an intriguing open question. We believe that positive results in this front might also shed light on the question of obtaining more efficient inference for large classes of properties, especially over small alphabets.

### Does adaptivity help?

This work does not provide any lower bounds for adaptive tests, and it will be interesting to do so; previously investigated properties like monotonicity yield an  $\Omega(d \log n)$  lower bound [13, 17], and we believe that “data flow” type properties, somewhat similar to our lower

## 11:12 Testing Local Properties of Arrays

bound constructions, can provide instances of 2-local properties that require at least  $n^c$  queries, for some constant  $c \leq 1$ , for the adaptive two-sided case.

However, it is not clear whether better lower bounds (even bounds of the type  $\Omega(n^{1+c})$ ) exist. It will be very interesting to prove better upper and lower bounds for testing local properties. Our conjecture is that *any* 2-local property is testable in  $n^{1+o(1)}g(d)$  queries (where  $g(d)$  depends only on  $d$ ), but proving a statement of this type might be very difficult.

### Using the unrepairability framework in other contexts

In this work we show that the concept of unrepairability allows to unify and reprove many property testing results on one-dimensional arrays. What about multi-dimensional arrays? for example, can one generalize the currently known proofs for “bounded derivative” properties (including monotonicity and Lipschitz continuity) in  $d$  dimensions to a larger class of local properties?

### Inference

As mentioned in Subsection 1.4.1, our test queries boundaries of block-like structures, and later infers whether each block is *unrepairable* (recall the definition from Subsection 1.4.1). The inference takes place without making any additional queries, and is based only on the property  $\mathcal{P}$ , the alphabet  $\Sigma$ , and the values of  $A$  in the boundary of the block.

The running time of the inference step is very large in general (although, as we have seen, in the 1D case it can be significantly improved using dynamic programming). The naive way to run the inference is by enumerating over all possible ways to fill the interior of the block, and checking whether each such possibility is indeed  $\mathcal{F}$ -free. The running time of this method is of order  $|\Sigma|^{O(n^d)}$  in general for  $d > 1$ , and is exponential in  $n$  even if  $|\Sigma| = 2$ .

However, for many natural properties, inference can be done much more efficiently. For example, in monotonicity testing, the inference amounts to checking that no pair of boundary entries violates the monotonicity. Our lower bound constructions depict other properties where inference is efficient: it is not hard to show that the running time of inference in both cases is  $O(k\epsilon^{-1/d}n^{d-1})$ , which is sublinear in  $n^d$  for a wide range of parameters.

Thus, we believe that understanding inference better – including tasks such as characterizing properties in which inference can be done efficiently, and understanding the inference time of specific properties of interest – would be an interesting direction for future research.

## 1.7 Property testing notation

The property testing notation we use along the paper is standard. Given a property  $\mathcal{P}$  of  $[n]^d$ -arrays over  $\Sigma$ , a proximity parameter  $\epsilon > 0$ , and query access to an unknown  $[n]^d$ -array  $A$ , a *two-sided error  $\epsilon$ -test* must accept  $A$  with probability at least  $2/3$  if  $A$  satisfies  $\mathcal{P}$ , and reject with probability  $2/3$  if  $A$  is  $\epsilon$ -far from  $\mathcal{P}$  (meaning that the relative *Hamming distance* of  $A$  from  $\mathcal{P}$  is at least  $\epsilon$ , that is, we need to modify at least  $\epsilon n^d$  values in  $A$  to make it satisfy  $\mathcal{P}$ ). A *one-sided error test* is defined similarly, but it must accept if  $A$  satisfies  $\mathcal{P}$ . A test is *non-adaptive* if it makes all of its queries in advance (prior to receiving any of the queried values), and *adaptive* otherwise.

### Organization

In Sections 2, 3 and 4 we prove the upper bounds: Section 2 is devoted to the infrastructure needed for the proof, Section 3 presents a simple but non-optimal test, and finally, Section 4 presents the optimal test and proves Theorems 1 and 2.

Due to space considerations, the proofs of the lower bounds are given in Sections 5 and 6 of the full version of this paper [5].

## 2 The grid structure

In this section we present the grid-like structure in  $[n]^d$  that we utilize for our tests.

► **Definition 5 (Interval partition).** A subset  $I \subseteq [n]$  is an *interval* if its elements are consecutive, that is, if  $I = \{x, x+1, \dots, x+y\}$  for some  $x \in [n]$  and  $y \geq 0$ . For any  $\ell \geq 0$ , we denote the set of the smallest  $\ell$  elements of  $I$  by  $I[:\ell]$  and also define  $I[\ell+1:] = I \setminus I[:\ell]$ . In the degenerate case that  $|I| < \ell$ , we define  $I[:\ell]$  to be equal to  $I$ .

For  $1 \leq w \leq n$ , an  $(n, w)$ -interval partition is a partition of  $[n]$  into a collection of disjoint intervals  $\mathcal{I} = (I_1, \dots, I_t)$  where the number of elements in each interval  $I_i$  is either  $w$  or  $w+1$ , and for any  $i < j$ , all elements of  $I_i$  are smaller than those in  $I_j$ .

► **Lemma 6.** For any positive integer  $n$  and  $0 \leq i \leq \log n$ , there exists an  $(n, \lfloor n/2^i \rfloor)$ -interval partition  $\mathcal{I}_i$  containing exactly  $2^i$  intervals, so that the family  $\{\mathcal{I}_i\}_{i=0}^{\lfloor \log n \rfloor}$  satisfies the following. For any  $i > j$  and interval  $I \in \mathcal{I}_i$ , there exists an interval  $I' \in \mathcal{I}_j$  satisfying  $I \subseteq I'$ .

**Proof.** For any  $i$  define  $n_i = \lfloor n/2^i \rfloor$ ; observe that  $n_0 = n$  and  $n_{i+1} = \lfloor n_i/2 \rfloor$  for any  $i$ . We prove the lemma by induction on  $i$ , starting by defining  $\mathcal{I}_0 = ([n])$ . Given  $\mathcal{I}_i = (I_1^i, \dots, I_{2^i}^i)$  in which all intervals are of length  $n_i$  or  $n_i+1$ , we define  $\mathcal{I}_{i+1}$  as follows. Each  $I_j^i \in \mathcal{I}_i$  is decomposed into two intervals  $I_{2j-1}^{i+1}, I_{2j}^{i+1}$  where  $|I_{2j-1}^{i+1}|, |I_{2j}^{i+1}| \in \{n_{i+1}, n_{i+1}+1\}$ , and all elements of  $I_{2j-1}^{i+1}$  are smaller than all elements of  $I_{2j}^{i+1}$ ; observe that such a decomposition is indeed always possible. Now define  $\mathcal{I}_{i+1} = (I_1^{i+1}, \dots, I_{2^{i+1}}^{i+1})$ . Clearly, the intervals of  $\mathcal{I}_{i+1}$  satisfy the last condition of the lemma. ◀

In particular, we conclude that for any positive integer  $w$  and any  $n \geq w$  there exists an integer  $w/2 \leq w' \leq w$  for which an  $(n, w')$ -interval partition exists.

► **Definition 7 ( $(n, d, k, w)$ -grid).** Let  $2 \leq w \leq n$  be integers for which an  $(n, w)$ -interval partition  $\mathcal{I} = (I_1, \dots, I_t)$  exists. For integers  $2 \leq k \leq w$  and  $d \geq 1$ , the  $(d)$ -dimensional  $(n, d, k, w)$ -grid induced by  $\mathcal{I}$  is the set

$$G = \left\{ (x_1, \dots, x_d) \in [n]^d \mid \exists i \in [d] \text{ such that } x_i \in \bigcup_{j=1}^t I_j[:k-1] \right\}.$$

We denote the family of all  $(n, d, k, w)$ -grids by  $\mathcal{G}(n, d, k, w)$ . As we have seen in Lemma 6, the family  $\mathcal{G}(n, d, k, w)$  is non-empty for any  $w = \lfloor n/2^i \rfloor$  satisfying  $w \geq k$ .

► **Definition 8 ( $G$ -block, Boundary, Closure).** Two tuples  $x = (x_1, \dots, x_d)$  and  $y = (y_1, \dots, y_d)$  in  $[n]^d$  are considered *neighbors* if  $\sum_{i=1}^d |x_i - y_i| = 1$ . Given a grid  $G \in \mathcal{G}(n, d, k, w)$ , consider the neighborhood graph of non-grid entries, i.e., the graph whose set of vertices is  $V = [n]^d \setminus G$  and two entries are connected if they are neighbors. A  $G$ -block  $B$  is a connected component of this graph, and the *closure* of  $B$  is

$$\overline{B} = \left\{ (x_1, \dots, x_d) \in [n]^d \mid \exists (y_1, \dots, y_d) \in B \text{ such that } \forall i \in [d] |x_i - y_i| < k \right\}.$$

Note that  $B \subseteq \overline{B}$ . Define the *boundary* of the block  $B$  as  $\partial B = \overline{B} \setminus B$ .



## 11:14 Testing Local Properties of Arrays

The above notions can naturally be defined with Cartesian products. Recall that the Cartesian product of sets  $X_1, \dots, X_d$ , denoted  $\prod_{j=1}^d X_j$  or  $X_1 \times \dots \times X_d$ , is the set of all tuples  $(x_1, \dots, x_d)$  with  $x_j \in X_j$  for any  $j \in [d]$ . Let  $G \in \mathcal{G}(n, d, k, w)$  be the grid induced by the interval partition  $\mathcal{I} = (I_1, \dots, I_t)$ . It is not difficult to verify that any  $G$ -block  $B$  can be defined as a Cartesian product  $B = \prod_{j=1}^d I_{i_j}[k:]$  for some intervals  $I_{i_1}, \dots, I_{i_d} \in \mathcal{I}$  (not necessarily different).

$\overline{B}$  and  $\partial B$  can also be defined accordingly, as we detail next. For  $k$  as above, define  $\overline{I_i} = I_i \cup I_{i+1}[k-1]$  for any  $1 \leq i \leq t$ , where we take  $I_{t+1} = \emptyset$  for consistency. Also define  $\partial I_i = \overline{I_i} \setminus I_i[k:] = I_i[k-1] \cup I_{i+1}[k-1]$ . With these in hand, we have

$$B = \prod_{j=1}^d I_{i_j}[k:]; \quad \overline{B} = \prod_{j=1}^d \overline{I_{i_j}}; \quad \partial B = \bigcup_{j=1}^d \overline{I_{i_1}} \times \dots \times \overline{I_{i_{j-1}}} \times \partial I_{i_j} \times \overline{I_{i_{j+1}}} \times \dots \times \overline{I_{i_d}} \quad (1)$$

Recall that  $|I_{i_j}| \in \{w, w+1\}$  for any  $j$ , implying that  $|I_{i_j}[k:]| \leq w+2-k$  and  $|\overline{I_{i_j}}| \leq w+k$ . Also note that  $|\partial I_{i_j}| \leq 2(k-1)$ . Thus,

$$|B| \leq (w+2-k)^d; \quad |\overline{B}| \leq (w+k)^d; \quad |\partial B| \leq 2d(k-1) \cdot (w+k)^{d-1}, \quad (2)$$

where the inequality on  $|\partial B|$  holds since each set in the union expression in (1) is of size at most  $(2k-2)(w+k)^{d-1}$ .

The following observation is a direct consequence of (1).

► **Observation 9.** *Let  $G \in \mathcal{G}(n, d, k, w)$ . The boundary of any  $G$ -block is contained in  $G$ .*

► **Lemma 10.** *For any  $G \in \mathcal{G}(n, d, k, w)$ , any width- $k$  subarray of an  $[n]^d$ -array intersects exactly one  $G$ -block  $B$ . Moreover, the subarray is contained in  $\overline{B}$ .*

**Proof.** Let  $\mathcal{I} = (I_1, \dots, I_t)$  be the interval partition inducing  $G$ . Suppose that the subarray  $S$  is in location  $(a_1, \dots, a_d)$  where  $a_j \in I_{i_j}$  for some  $i_1, \dots, i_d$  not necessarily distinct. In other words, the set of entries in  $S$  is  $\prod_{j=1}^d S_j$  where  $S_j = \{a_j, a_j+1, \dots, a_j+k-1\}$  for any  $j \in [d]$ . We argue that  $S$  is contained in  $\overline{B}$ , where  $B = I_{i_1}[k:] \times \dots \times I_{i_d}[k:]$ : The fact that  $a_j \in I_{i_j}$  implies that  $a_j+1, \dots, a_j+k-1 \in I_{i_j} \cup I_{i_j+1}[k-1]$ . It follows from (1) that  $S \subseteq \overline{B}$ . From Observation 9 we conclude that  $S$  does not intersect any block other than  $B$ , and it remains to show that  $S$  intersects  $B$ . Indeed, for any  $1 \leq j \leq d$ , the fact that  $a_j \in I_{i_j}$  implies that one of the elements  $a_j, \dots, a_j+k-1$  must be contained in  $I_{i_j}[k:]$ . Denoting this element by  $b_j$ , we conclude that  $(b_1, \dots, b_d) \in S \cap B$ . ◀

### 3 Testing with grid queries

In this section we prove the following upper bound for all  $k$ -local properties; its proof serves as a warm-up towards proving the main upper bound of Theorem 1.

► **Theorem 11.** *Any  $k$ -local property of  $[n]^d$ -arrays over any alphabet is  $\epsilon$ -testable with one-sided error using no more than  $2(d+1)n^{d-\frac{d}{d+1}}k^{\frac{d}{d+1}}\epsilon^{-\frac{1}{d+1}}$  non-adaptive queries.*

The upper bound of Theorem 11 is sublinear in the size of the array as long as  $k/\epsilon^{1/d} = o(n)$ . The rest of the section is dedicated to the proof of the theorem. We may assume that  $k \leq \epsilon^{1/d}n/4$ , as otherwise the expression in the statement of the theorem is larger than  $n^d$  and the proof follows trivially by querying all  $[n]^d$  entries of the given input array. Under this assumption, it holds that  $2k \leq n^{d/(d+1)}k^{1/(d+1)}\epsilon^{1/(d+1)}$ .



► **Definition 12** (Unrepairable block). Let  $A$  be an  $[n]^d$ -array over  $\Sigma$ , and let  $G \in \mathcal{G}(n, d, k, w)$ . A  $G$ -block  $B$  is  $(\mathcal{P}, A)$ -unrepairable (or simply *unrepairable*, if  $\mathcal{P}$  and  $A$  are clear from context) if any  $[n]^d$ -array  $A'$  over  $\Sigma$  that satisfies  $A'(x) = A(x)$  for any  $x \in \partial B$ , including the case  $A' = A$ , contains an  $\mathcal{F}$ -copy in  $\overline{B}$ . Otherwise, the block  $B$  is said to be  $(\mathcal{P}, A)$ -repairable.

Note that the (un)repairability of a block  $B$  is determined solely by the values of  $A$  on  $\partial B$ , and that an unrepairable block always contains an  $\mathcal{F}$ -copy. These two facts inspire the following lemma, which serves as the conceptual core behind the test of Theorem 11.

► **Lemma 13.** *Suppose that  $A$  is an  $[n]^d$ -array that is  $\epsilon$ -far from satisfying a  $k$ -local property  $\mathcal{P}(\mathcal{F})$ , and let  $G \in \mathcal{G}(n, d, k, w)$  where  $w \geq k$ . Then at least one of the following holds.*

- *There exists a  $(\mathcal{P}, A)$ -unrepairable  $G$ -block.*
- *For at least an  $\epsilon$ -fraction of the  $G$ -blocks  $B$ , there is an  $\mathcal{F}$ -copy in  $\overline{B}$ .*

**Proof.** Suppose that the first condition does not hold, that is, all  $G$ -blocks are  $(\mathcal{P}, A)$ -repairable. By Lemma 10, every  $\mathcal{F}$ -copy is contained in the closure of some  $G$ -block.

Let  $\mathcal{C}$  denote the collection of all  $G$ -blocks  $B$  such that  $A$  contains an  $\mathcal{F}$ -copy in  $\overline{B}$ . By the repairability, the values of  $A$  in each block  $B \in \mathcal{C}$  can be modified so that after the modification,  $A$  will not contain an  $\mathcal{F}$ -copy in  $\overline{B}$ . We stress that the modifications for each block  $B$  appear only in  $B$  itself and do not modify entries on the grid, so by Lemma 10, they cannot create new  $\mathcal{F}$ -copies in the closure of other blocks.

After applying all of the above modifications to  $A$ , we get an  $\mathcal{F}$ -free array, i.e., an array that satisfies  $\mathcal{P}$ .  $A$  was initially  $\epsilon$ -far from  $\mathcal{P}$ , and the number of entries in each block is bounded by  $(w + 2 - k)^d \leq w^d$ , implying that at least an  $\epsilon$ -fraction of the blocks belong to  $\mathcal{C}$ . ◀

**Proof of Theorem 11.** We may assume that  $k^d/\epsilon \leq n^d/2$ , otherwise our test may trivially query all  $n^d$  entries of  $A$ . Our (non-adaptive) test  $T$  picks  $W = \lfloor n^{d/(d+1)} k^{1/(d+1)} \epsilon^{1/(d+1)} \rfloor \geq 2k$ , and an integer  $w$  satisfying  $k \leq W/2 \leq w \leq W$ , for which an  $(n, w)$ -interval partition exists.  $T$  now makes the following queries.

1.  $T$  queries all entries of an arbitrarily chosen grid  $G \in \mathcal{G}(n, d, k, w)$ . The number of entries in any grid is at most  $dn^d(k-1)/w \leq 2dn^{d-\frac{d}{d+1}} k^{\frac{d}{d+1}} \epsilon^{-\frac{1}{d+1}}$ .
2.  $T$  chooses a collection  $\mathcal{B}$  of  $2/\epsilon$   $G$ -blocks uniformly at random and queries all entries in these blocks. Since each block contains at most  $(w + 2 - k)^d \leq W^d$  entries, the total number of queries is bounded by  $2W^d/\epsilon \leq 2n^{d-\frac{d}{d+1}} k^{\frac{d}{d+1}} \epsilon^{-\frac{1}{d+1}}$ . Note that the boundaries of all blocks are queried in the first step (since they are contained in the grid). Thus, for any block  $B \in \mathcal{B}$ , the test queries all entries of  $\overline{B}$ .

The total number of queries in the above two steps is  $2(d+1)n^{d-\frac{d}{d+1}} k^{\frac{d}{d+1}} \epsilon^{-\frac{1}{d+1}}$ .

After querying all entries of the grid (and in particular, the whole boundaries of all of the blocks),  $T$  can determine for every  $G$ -block  $B$  whether it is  $(\mathcal{P}, A)$ -unrepairable or not.  $T$  rejects if at least one of the blocks is unrepairable or if it found an  $\mathcal{F}$ -copy in  $\overline{B}$  for some  $B \in \mathcal{B}$ , and accepts otherwise. The test has one-sided error, since an unrepairable block must contain an  $\mathcal{F}$ -copy. In view of Lemma 13,  $T$  rejects arrays  $A$  that are  $\epsilon$ -far from  $\mathcal{P}$  with probability at least  $2/3$ : If  $A$  satisfies the first condition of Lemma 13, then  $T$  always rejects. If the second condition holds, the probability that none of the  $2/\epsilon$  closures  $\overline{B}$  for  $B \in \mathcal{B}$  contains an  $\mathcal{F}$ -copy is bounded by  $(1 - \epsilon)^{2/\epsilon} < e^{-2}$ , so  $T$  rejects with probability at least  $1 - e^{-2} > 2/3$ . ◀

#### 4 Systems of grids and testing with spherical queries

In this section we prove Theorems 1 and 2. We do so by considering a system of grids with varying block sizes, defined as follows.

► **Definition 14.** Let  $d > 0$  and  $2 \leq k \leq w \leq n$  be integers. An  $(n, d, k, w)$ -system of grids is an  $(r + 1)$ -tuple  $(G_0, G_1, \dots, G_r)$  of grids, for  $r(n, w) = \lfloor \log(n/w) \rfloor$ , so that

- $G_i \in \mathcal{G}(n, d, k, \lfloor n/2^{r-i} \rfloor)$  for any  $0 \leq i \leq r$ .
- $G_0 \supseteq G_1 \supseteq \dots \supseteq G_r$  (as subsets of  $[n]^d$ ). In particular, for any  $i < j \leq r$ , any  $G_i$ -block  $B$  is contained in a  $G_j$ -block  $B'$ , and we say that  $B'$  is an *ancestor* of  $B$ . Specifically, the  $G_{i+1}$ -block containing  $B$  is called the *parent* of  $B$  and denoted by  $\text{Par}(B)$ . For the only  $G_r$ -block,  $B_r$ , we define  $\text{Par}(B_r)$  as the whole domain  $[n]^d$ .

$r(n, w)$  was chosen so that  $w \leq n/2^r < 2w$ , making  $G_0$  a  $\mathcal{G}(n, d, k, w')$ -grid for  $w \leq w' < 2w$ . As we shall see, when working with such a system, unrepairability of blocks can be handled in a query-efficient way. The following lemma asserts that such a system of grids exists for any suitable choice of parameters.

► **Lemma 15.** An  $(n, d, k, w)$ -system of grids exists for all  $d > 0$  and  $2 \leq k \leq w \leq n$ .

**Proof.** Consider the family of interval partitions  $\mathcal{I}_0, \dots, \mathcal{I}_{\lfloor \log n \rfloor}$  obtained by Lemma 6. For each  $0 \leq i \leq r(n, w)$  define  $G_i$  as the  $(n, d, k, \lfloor n/2^{r-i} \rfloor)$ -grid induced by  $\mathcal{I}_{r-i}$ . It is not hard to verify that  $(G_0, \dots, G_r)$  satisfies all requirements of an  $(n, d, k, w)$ -system of grids. ◀

For the rest of the section, fix a  $k$ -local property  $\mathcal{P}(\mathcal{F})$  of  $[n]^d$ -arrays over  $\Sigma$ , as well as an  $[n]^d$ -array  $A$  over  $\Sigma$ . Consider an  $(n, d, k, w)$ -system of grids  $(G_0, \dots, G_r)$  constructed as described in the proof of Lemma 15, where  $w$  will be determined later. (For now it suffices to require, as usual, that  $2 \leq k \leq w \leq n$ .)

We say that a  $G_i$ -block  $B$  is a  $(\mathcal{P}, A)$ -witness if one of the following holds.

- $i = 0$  and the array  $A$  contains an  $\mathcal{F}$ -copy in the closure  $\overline{B}$ .
- $i > 0$  and  $B$  is  $(\mathcal{P}, A)$ -unrepairable.

Recall that the closure of unrepairable blocks cannot be  $\mathcal{F}$ -free, so the closure of any witness block contains an  $\mathcal{F}$ -copy. We say that a witness block  $B$  is *maximal* if all of its ancestors are not witnesses, that is, they are repairable.

► **Observation 16.** Any  $(\mathcal{P}, A)$ -witness is contained in a maximal  $(\mathcal{P}, A)$ -witness.

We define the *maximal witness family*  $\mathcal{W}$  as the set of all maximal  $(\mathcal{P}, A)$ -witness blocks. Obviously, the blocks in  $\mathcal{W}$  might come from different  $G_i$ 's

► **Observation 17.**  $B_1 \cap B_2 = \emptyset$  for any two blocks  $B_1, B_2 \in \mathcal{W}$ .

► **Lemma 18.** All  $\mathcal{F}$ -copies in  $A$  are fully contained in  $\bigcup_{B \in \mathcal{W}} \overline{B}$ .

**Proof.** Let  $F$  be an  $\mathcal{F}$ -copy in  $A$ . By Lemma 10,  $F$  is contained in the closure of a unique  $G_0$ -block  $B_F$ ; hence,  $B_F$  is a  $(\mathcal{P}, A)$ -witness. From Observation 16 we have  $B_F \subseteq B'$  for some maximal  $(\mathcal{P}, A)$ -witness  $B'$ . We conclude that  $F \in \overline{B_F} \subseteq \overline{B'}$ . ◀

► **Lemma 19.** One can make  $A$  satisfy  $\mathcal{P}$  by only modifying entries of  $A$  in  $\bigcup_{B \in \mathcal{W}} \text{Par}(B)$ .

**Proof.** Fix  $B \in \mathcal{W}$ .  $B$  is a maximal  $(\mathcal{P}, A)$ -witness, so  $\text{Par}(B)$  is repairable.<sup>1</sup> Thus, One can make  $\overline{\text{Par}(B)}$   $\mathcal{F}$ -free by only modifying entries inside  $\text{Par}(B)$ . By Lemma 10,

<sup>1</sup> Note that when  $B = B_r$  is the maximal witness considered,  $\text{Par}(B)$  is  $[n]^d$ ; the latter is repairable for any non-empty property.

width- $k$  subarrays that are not fully contained in  $\overline{\text{Par}(B)}$  are left unchanged. Therefore, this modification does not create any new  $\mathcal{F}$ -copies in  $A$ . Seeing that all  $\mathcal{F}$ -copies in  $A$  are originally contained in  $\bigcup_{B \in \mathcal{W}} \overline{B} \subseteq \bigcup_{B \in \mathcal{W}} \overline{\text{Par}(B)}$ , applying these modifications for all  $B \in \mathcal{W}$  deletes all  $\mathcal{F}$ -copies in  $A$  without creating new ones, so in the end of the process  $A$  satisfies  $\mathcal{P}$ .  $\blacktriangleleft$

We may assume that  $k \leq \epsilon^{1/d}n/10$ , as otherwise the expression in the theorem is  $\Omega(n^d)$ . We choose  $w = 2k$ , working with an  $(n, d, k, 2k)$ -system of grids from now on. A very useful consequence of this choice of  $w$  is that here the parent of a block  $B$  cannot be much larger than  $B$  itself.

► **Lemma 20.** *Let  $(G_0, G_1, \dots, G_r)$  be an  $(n, d, k, 2k)$ -system of grids. Then for any  $0 \leq i \leq r$  and any  $G_i$ -block  $B$  it holds that  $|\text{Par}(B)|/|B| < 3^d$ .*

**Proof.** For  $i = r$  this is trivial. Now fix  $i < r$  and let  $B$  be a  $G_i$ -block. Recall that, following (1), one can write  $B = \prod_{j=1}^d I_{i_j}[k:]$  where each interval  $I_{i_j}$  (for  $j \in [d]$ ) is of size at least  $2k \geq 4$ . On the other hand, we can also write  $\text{Par}(B) = \prod_{j=1}^d I'_{i'_j}[k:]$  where  $I'_{i'_j} \supseteq I_{i_j}$  for any  $j \in [d]$ . It is not hard to verify that  $|I'_{i'_j}| \leq 2|I_{i_j}| + 1$  most hold, and so

$$\frac{|\text{Par}B|}{|B|} = \prod_{j=1}^d \frac{|I'_{i'_j}| - (k-1)}{|I_{i_j}| - (k-1)} \leq \prod_{j=1}^d \frac{2|I_{i_j}| + 1 - (k-1)}{|I_{i_j}| - (k-1)} \leq \left( \frac{2 \cdot 2k - k + 2}{2k - k + 1} \right)^d < 3^d$$

where the second inequality holds since  $|I_{i_j}| \geq 2k$  for any  $j$ .  $\blacktriangleleft$

The next corollary follows immediately from Lemmas 19 and 20.

► **Corollary 21.** *Suppose that  $A$  is  $\epsilon$ -far from  $\mathcal{P}$ . Then the total number of entries in the blocks of  $\mathcal{W}$  is at least  $\epsilon(n/3)^d$ .*

We are now ready for the proof of the main upper bound of this paper, Theorem 1.

## 4.1 Proof of Theorem 1

As before, we may assume that  $k \leq \epsilon^{1/d}n/10$ . For larger  $k$ , the expression in the theorem dominates  $n^d$  and thus becomes trivial. Consider the  $(n, d, k, 2k)$ -system of grids  $(G_0, G_1, \dots, G_r)$  mentioned above. For any  $0 \leq i \leq r$ , define  $\delta_i = |\mathcal{B}_i \cap \mathcal{W}|/|\mathcal{B}_i|$ , where  $\mathcal{B}_i$  is the set of all  $G_i$ -blocks. In other words,  $\delta_i$  is the fraction of maximal witnesses among the  $G_i$ -blocks. By Corollary 21, if  $A$  is  $\epsilon$ -far from  $\mathcal{P}$  then  $\sum_{i=0}^r \delta_i \geq \epsilon/3^d$ . Define  $r' = \lfloor \log(\epsilon^{1/d}n/k) \rfloor \geq 1$ , noting that  $G_{r'} \in \mathcal{G}(n, d, k, w_{r'})$  with  $w_{r'} \geq 2k \cdot 2^{r'} \geq \epsilon^{1/d}n$ . Thus, the total number of blocks in  $\mathcal{B}_{r'}$  is bounded by  $(n/w_{r'})^d \leq 1/\epsilon$ .

### The test

We iterate the following basic step  $2 \cdot 3^d/\epsilon$  times.

1. Pick  $B \in \mathcal{B}_0$  uniformly at random and query all entries of  $\overline{B}$ .
  2. For any  $1 \leq i \leq r'$ , pick  $B \in \mathcal{B}_i$  uniformly at random and query all entries of  $\bigcup_{B \in \mathcal{Q}_0} \partial B$ .
- Finally, the test rejects if and only if at least one of the blocks  $B$  picked during the process is a  $(\mathcal{P}, A)$ -witness. (Recall that querying all boundary entries of a  $G_i$ -block for  $i > 0$  suffices to determine whether it is unrepairable, and thus a witness.)

The test is clearly non-adaptive, and has one-sided error: It only rejects if it finds a witness. As we have seen earlier, all witnesses contain an  $\mathcal{F}$ -copy.

## 11:18 Testing Local Properties of Arrays

The test is *canonical* in the following sense. The choice of queries in every basic step depends only on  $n, d, k$ , and (weakly) on  $\epsilon$ , and is independent of the property  $\mathcal{P}$  or the alphabet  $\Sigma$ . To determine which entries constitute a block, it suffices to know the parameters of the block, that depend only on  $n, d, k$ ; the dependence in  $\epsilon$  is only taken into account in the choice of  $r'$ . The test only considers  $\mathcal{P}$  in order to determine whether each queried block is a witness.

### Analysis

Suppose that  $A$  is  $\epsilon$ -far from  $\mathcal{P}$ . If  $\delta_i > 0$  for some  $i > r'$  then it must hold that  $\delta_{r'} > 0$  as well (since any unrepairable  $G_i$ -block must contain an unrepairable  $G_{i'}$ -block for any  $i' < i$ ). By the choice of  $r'$ , we must have  $\delta_{r'} \geq 1/|\mathcal{B}_{r'}| \geq \epsilon$  in this case. If the above doesn't hold, then  $\delta_i = 0$  for any  $i \geq r'$ , implying that  $\sum_{i=0}^{r'-1} \delta_i \geq \epsilon/3^d$ . Therefore, in both cases, we have  $\sum_{i=0}^{r'} \delta_i \geq \epsilon/3^d$ .

The probability that a random  $\mathcal{B}_i$ -block is a witness is at least  $\delta_i$ , and therefore the probability that a single basic step leads to a rejection of  $A$  is at least  $\sum_{i=0}^{r'} \delta_i \geq \epsilon/3^d$ . Running  $2 \cdot 3^d/\epsilon$  independent iterations of the basic step ensures that the test will accept  $A$  with probability at most  $(1 - \epsilon/3^d)^{2 \cdot 3^d/\epsilon} \leq e^{-2} < 2/3$ , as desired.

### Query complexity

For  $d = 1$ , the query complexity of each basic step is  $O(kr')$ : The test queries  $\overline{B}$  for a single block  $B \in \mathcal{B}_0$ , and the boundaries of  $r'$  larger blocks. Considering the parameters of our system of grids, we have  $|B| \leq 4k$  and so  $|\overline{B}| < 6k$ . On the other hand, the boundary of each of the larger blocks is of size at most  $2k - 2$ . Therefore, the total query complexity for the 1D test is  $O(kr'/\epsilon) = O\left(\frac{k}{\epsilon} \log(\epsilon n/k)\right)$  as desired.

For  $d > 1$ , consider a single basic step, and for any  $0 \leq i \leq r'$  let  $B_i \in \mathcal{B}_i$  be the  $\mathcal{G}_i$ -block picked in this step. From (2) we have  $|\overline{B}_0| \leq (6k)^d$ , while for any  $i > 0$  we have  $|\partial B_i| \leq 2d(k-1)(4k \cdot 2^i + k)^{d-1} = O(d \cdot (4k)^d \cdot 2^{(d-1)i})$ . Note that the last expression grows exponentially with  $(d-1)i$ , so the total number of queries in a single basic step is  $O((6k)^d + d \cdot (4k)^d 2^{(d-1)r'})$ . Plugging in  $r'$ , we have  $2^{(d-1)r'} = \frac{\epsilon^{(d-1)/d}}{k^{d-1}} n^{d-1}$ . As the test runs  $O(3^d/\epsilon)$  iterations of the basic step, we conclude that the total query complexity is bounded by  $c^d k \epsilon^{-1/d} n^{d-1}$  for an absolute constant  $c > 0$ , completing the proof of Theorem 1.

## 4.2 Proximity oblivious test

The proof of Theorem 2 follows by a very simple modification of the proof of Theorem 1. The desired proximity oblivious test (POT) is the so called “basic step” from the above test, with  $r$  replacing  $r'$  (since  $r'$  depends on  $\epsilon$ ). The POT rejects if it infers that one of the blocks queried is a witness, like the above test. Its query complexity is  $O(kr) = O(k \log n/k)$  for  $d = 1$ . In the case  $d > 1$ , the query complexity is dominated by the size of  $\partial B_r$ , which is bounded by  $O(dkn^{d-1})$ .

Clearly this POT has one-sided error, and its queries do not depend on the property  $\mathcal{P}$  and the alphabet  $\Sigma$  (on the other hand, they do depend on  $n, d, k$ ). Using the notation of the previous subsection and denoting by  $\epsilon_A$  the Hamming distance of a given input  $A$  from  $\mathcal{P}$ , we get (exactly as in the beginning of Subsection 4.1) a rejection probability of at least  $\sum_{i=0}^r \delta_i \geq \epsilon_A/3^d$  for  $A$ , which is linear in  $\epsilon_A$  for fixed  $d$ . This concludes the proof.

---

**References**


---


- 1 N. Alon, O. Ben-Eliezer, and E. 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.
- 2 N. Alon, M. Krivelevich, I. Newman, and M. Szegedy. Regular languages are testable with a constant number of queries. *SIAM Journal on Computing*, 30:1842–1862, 2001.
- 3 P. Awasthi, M. Jha, M. Molinaro, and S. Raskhodnikova. Testing Lipschitz functions on hypergrid domains. *Algorithmica*, 74:1055–1081, 2016.
- 4 A. Belovs. Adaptive Lower Bound for Testing Monotonicity on the Line. In *Approximation, Randomization and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM)*, pages 31:1–31:10, 2018.
- 5 O. Ben-Eliezer. Testing local properties of arrays. *Electronic Colloquium on Computational Complexity (ECCC)*, 2018. full version. URL: <https://eccc.weizmann.ac.il/report/2018/196/>.
- 6 O. Ben-Eliezer and E. Fischer. Earthmover Resilience and Testing in Ordered Structures. In *Proceedings of the 33rd Conference on Computational Complexity (CCC)*, pages 18:1–18:35, 2018.
- 7 O. Ben-Eliezer, S. Korman, and D. Reichman. Deleting and Testing Forbidden Patterns in Multi-Dimensional Arrays. In *Proceedings of the 44th International Colloquium on Automata, Languages and Programming (ICALP)*, pages 9:1–9:14, 2017.
- 8 P. Berman, M. Murzabulatov, and S. Raskhodnikova. Testing convexity of figures under the uniform distribution. In *Proceedings of the 32nd International Symposium on Computational Geometry (SoCG)*, pages 17:1–17:15, 2016.
- 9 P. Berman, M. Murzabulatov, and S. Raskhodnikova. Tolerant testers of image properties. In *Proceedings of the 43rd International Colloquium on Automata, Languages and Programming (ICALP)*, pages 90:1–90:14, 2016.
- 10 P. Berman, S. Raskhodnikova, and G. Yaroslavtsev.  $L_p$ -testing. In *Proceedings of the 46th ACM Symposium on the Theory of Computing (STOC)*, pages 164–173, 2014.
- 11 E. Blais and A. Bommireddi. Testing submodularity and other properties of valuation functions. In *Proceedings of the 8th Innovations in Theoretical Computer Science Conference (ITCS)*, pages 33:1–33:17, 2017.
- 12 E. Blais, J. Brody, and K. Matulef. Property testing lower bounds via communication complexity. *Computational Complexity*, 21:311–358, 2012.
- 13 E. Blais, S. Raskhodnikova, and G. Yaroslavtsev. Lower bounds for testing properties of functions over hypergrid domains. In *Proceedings of the 29th Conference on Computational Complexity (CCC)*, pages 309–320, 2014.
- 14 D. Chakrabarty. Monotonicity testing. *Encyclopedia of Algorithms*, pages 1352–1356, 2016.
- 15 D. Chakrabarty, K. Dixit, M. Jha, and C. Seshadhri. Property testing on Product Distributions: Optimal Testers for Bounded Derivative Properties. *ACM Transactions on Algorithms*, 13:20:1–20:30, 2017.
- 16 D. Chakrabarty and C. Seshadhri. Optimal bounds for monotonicity and Lipschitz testing over hypercubes and hypergrids. In *Proceedings of the 45th ACM Symposium on the Theory of Computing (STOC)*, pages 419–428, 2013.
- 17 D. Chakrabarty and C. Seshadhri. An optimal lower bound for monotonicity testing over hypergrids. *Theory of Computing*, 10:453–464, 2014.
- 18 X. Chen, A. Freilich, R. Servedio, and T. Sun. Sample-based high-dimensional convexity testing. In *Approximation, Randomization and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM)*, pages 37:1–37:20, 2017.
- 19 F. Ergün, S. Kannan, R. Kumar, R. Rubinfeld, and M. Viswanathan. Spot-checkers. *Journal of Computer and System Sciences*, 60:717–751, 2000.

- 20 E. Fischer. On the strength of comparisons in property testing. *Information and Computation*, 25:107–116, 2004.
- 21 O. Goldreich. *Introduction to property testing*. Cambridge University Press, 2017.
- 22 O. Goldreich, S. Goldwasser, and D. Ron. Property testing and its connection to learning and approximation. *Journal of the ACM*, 45:653–750, 1998.
- 23 O. Goldreich and D. Ron. On proximity-oblivious testing. *SIAM Journal on Computing*, 40:534–566, 2011.
- 24 M. Jha and S. Raskhodnikova. Testing and reconstruction of Lipschitz functions with applications to data privacy. *SIAM Journal on Computing*, 42:700–731, 2013.
- 25 S. Korman, D. Reichman, G. Tsur, and S. Avidan. Fast-Match: Fast Affine Template Matching. *International Journal of Computer Vision*, 121:111–125, 2017.
- 26 S. Moriguchi and K. Murota. On discrete Hessian matrix and convex extendability. *Journal of the Operations Research Society of Japan*, 55:48–62, 2012.
- 27 K. Murota. *Discrete Convex Analysis*. Society for Industrial and Applied Mathematics, 2003.
- 28 I. Newman and C. Sohler. Every property of hyperfinite graphs is testable. *SIAM Journal on Computing*, 42:1095–1112, 2013.
- 29 R. Pallavoor, S. Raskhodnikova, and N. Varma. Parameterized property testing of functions. *ACM Transactions on Computation Theory*, 9:17:1–17:19, 2018.
- 30 M. Parnas, D. Ron, and R. Rubinfeld. On testing convexity and submodularity. *SIAM Journal on Computing*, 32:1158–184, 2003.
- 31 L. Rademacher and S. Vempala. Testing geometric convexity. In *Foundations of Software Technology and Theoretical Computer Science Conference (FSTTCS)*, pages 469–480, 2004.
- 32 S. Raskhodnikova. Testing if an array is sorted. *Encyclopedia of Algorithms*, pages 2219–2222, 2016.
- 33 R. Rubinfeld and M. Sudan. Robust characterizations of polynomials with applications to program testing. *SIAM Journal on Computing*, 25:252–271, 1996.
- 34 C. Seshadhri and Jan Vondrák. Is Submodularity Testable? *Algorithmica*, 69:1–25, 2014.
- 35 A. C. Yao. Probabilistic computations: Toward a unified measure of complexity. In *Proceedings of the 18th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 222–227, 1977.

# The Complexity of User Retention


**Eli Ben-Sasson**

Department of Computer Science, Technion, Haifa, Israel  
eli@cs.technion.ac.il

 <https://orcid.org/0000-0002-0708-0483>

**Eden Saig**

Department of Computer Science, Technion, Haifa, Israel  
edens@cs.technion.ac.il

 <https://orcid.org/0000-0002-0810-2218>

---

## Abstract

This paper studies families of distributions  $\mathcal{T}$  that are amenable to *retentive learning*, meaning that an expert can *retain* users that seek to predict their future, assuming user attributes are sampled from  $\mathcal{T}$  and exposed gradually over time. Limited *attention span* is the main problem experts face in our model. We make two contributions.

First, we formally define the notions of *retentively learnable* distributions and properties. Along the way, we define a *retention complexity* measure of distributions and a natural class of *retentive scoring rules* that model the way users evaluate experts they interact with. These rules are shown to be tightly connected to truth-eliciting “proper scoring rules” studied in Decision Theory since the 1950’s [McCarthy, PNAS 1956].

Second, we take a first step towards relating retention complexity to other measures of significance in computational complexity. In particular, we show that linear properties (over the binary field) are retentively learnable, whereas random Low Density Parity Check (LDPC) codes have, with high probability, maximal retention complexity. Intriguingly, these results resemble known results from the field of property testing and suggest that deeper connections between retentive distributions and locally testable properties may exist.

**2012 ACM Subject Classification** Theory of computation → Models of computation

**Keywords and phrases** retentive learning, retention complexity, information elicitation, proper scoring rules

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2019.12

**Related Version** <https://ecc.weizmann.ac.il/report/2017/160/>

**Funding** This work was supported by the Israeli Science Foundation under grant number 1501/14.

**Acknowledgements** We thank Yuval Filmus for many helpful discussions. We thank the anonymous reviewers for their insightful comments on an earlier version of this paper.

## 1 Introduction

Certain aspects of life – child development, love, psychological disorders, etc. – seem comprehensible and somewhat predictable *only* when addressed jointly with an expert, via an interactive process that unfolds over time. This work initiates the formal study of phenomena that are amenable to an interactive *collaborative discovery* process between an expert and a layperson. This collaboration unfolds over time, as more aspects of the phenomena become apparent to the layperson. We model this by associating each layperson with a sequence



© Eli Ben-Sasson and Eden Saig;  
licensed under Creative Commons License CC-BY  
10th Innovations in Theoretical Computer Science (ITCS 2019).  
Editor: Avrim Blum; Article No. 12; pp. 12:1–12:30



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

$x = (x_1, \dots, x_n)$  sampled from some underlying distribution  $\mathcal{T}$ , and assume that attributes of the phenomena (entries of  $x$ ) are gradually exposed over time, in response to active querying by the layperson. The interactive process has the expert predicting the value of  $x_i$  before it is revealed to the layperson. At the time of revelation, the layperson also re-evaluates his assessment of the expert’s utility to him: outcomes that seem “obvious” and predictable to the user are viewed as less helpful than “surprising” and unpredictable ones. At the end of each step, the layperson must decide whether to terminate his relationship with the expert or continue with it.

A major factor in our study is the *limited attention span* that characterizes users; consequently, the expert’s goal is to *retain* the layperson by constantly supplying him with meaningful and surprising insights about the phenomena. The terms *guru* (instead of expert) and *follower* (as opposed to layperson) better capture this dynamic so we use these terms henceforth. The main advantage the guru has over her followers is a greater ability to understand the phenomena, which we model by assuming the guru has a greater *memory span* than her followers. Using these terms, we define a class of distributions to be *retentively learnable* if a fixed *constant* advantage in memory span suffices for a guru to retain her followers throughout the entire collaborative discovery process. Our main complexity result is that the property of *linearity* over the binary field is retentively learnable (Theorem 10), whereas arbitrary linear spaces are maximally non-retentive (Theorem 13). To state these results, we first have to define our model and study its intrinsic properties. These properties turn out to be non-trivial to study, and lead to surprising results that are of independent interest. In particular, our first main result (Theorem 4) is a non-intuitive characterization of retentive scoring rules in terms of *proper scoring rules* studied in Decision Theory.

## Roadmap

Subsection 1.1 further explains, informally, the kind of guru–follower dynamics that we aim to formalize and understand. Subsection 1.2 gives a formal description of the model. In Subsection 1.3, we discuss the concept of *retentive scoring*, which allows us to describe the collaborative discovery process in an incentive-compatible way, taking agent rationality into account. Subsection 1.4 adds the layer of *limited memory span* to characterize the discrepancies between the guru and follower, and between fellow gurus. In Subsection 1.5, we state our main complexity results for linear properties. Subsection 1.6 reviews the latest related work, and finally Subsection 1.7 summarizes the main contributions and questions to be explored in the future.

## 1.1 The Guru’s Problem

In today’s information society, crowd-based automated gurus gather data from users on a voluntary basis in order to produce meaningful insights. The quality of insights greatly depends on the amount and quality of data provided by the users, but those users have limited attention, giving rise to the study of *attention economy* [12, 16]. By asking “*interesting questions*” and making “*meaningful predictions*”, an automated guru can retain users, but only if it “knows” how to ask “interesting” questions and provide “meaningful” feedback.

The phenomenon that motivated this research is that of *early child development*; the gurus are experts in this field and the followers are parents of newborn babies [3]. For the sake of concreteness we shall continue using this particular setting to describe our model but it may be conveniently replaced by the reader with physicians or psychologists playing the gurus as they interact with patients (followers) regarding a complex medical or mental



problem, or with financial advisors as gurus and their follower clientele. In these and similar settings, gurus and followers discuss a complex phenomenon that evolves over time, which the followers wish to understand, and about which the guru claims to have an advantage of “*wisdom*” over them.

## 1.2 The Collaborative Discovery model

The *phenomenon* about which the guru and her followers interact is modeled by a *distribution*  $\mathcal{T}$  over  $\mathcal{X}^\Gamma$ , where  $\Gamma$  is the set of properties manifested by the phenomenon and  $\mathcal{X}$  is an arbitrary input space. The two input spaces mentioned in this paper are the binary space  $\mathcal{X} = \{0, 1\}$  and the finite categorical space  $\mathcal{X} = \{0, \dots, n\}$ . In the context of childhood development,  $\Gamma$  is the set of developmental milestones like “*first smile*”, and each follower (associated, for simplicity, with a parent of a single child) is represented by a sample  $u \in \mathcal{X}^\Gamma$  that describes the ages at which that child achieved each milestone. By time  $t$ , the follower discloses to the guru  $u|_{\Gamma_t}$ , the restriction of her sample  $u$  to a subset  $\Gamma_t \subseteq \Gamma$ . Additional attributes of  $u$  may be revealed later in time, others might be disclosed by the follower if prompted to do so, while certain attributes will remain forever latent.

The follower seeks the guru’s assistance in predicting “meaningful information” that is currently unknown to the follower. The guru and follower *interact* over a number of rounds but the follower will terminate the interaction if the guru is judged to be unhelpful (in a manner formalized below). During each round of interaction, the guru makes a prediction by announcing a distribution  $P_{\gamma_t}$  over  $\mathcal{X}$  that she claims is the true one for a latent attribute  $\gamma_t \notin \Gamma_t$ ; the follower has a distribution  $Q_{\gamma_t}$  that she believes corresponds to  $\gamma_t$ . (Modern gurus and followers are comfortable discussing probabilities rather than predicting a single event as is the case with pre-election polling results.) The way  $\gamma_t$  is selected from  $\Gamma \setminus \Gamma_t$  and its effect on the process is left to future work. The follower now queries  $\gamma_t$  and reports the true value, denoted  $u_{\gamma_t}$ , which is derived from Nature’s “true” distribution. After each round the follower updates the strength of her *retention* by the guru. We assume this strength is given by a *retention parameter*  $r_t$  that starts with a fixed value  $r_0$  and varies with time; once  $r_t$  turns negative the follower will be said to have lost all faith in the guru and therefore terminate the interaction. The main objective of the guru is to maintain  $r_t \geq 0$  for all  $t \geq 0$ ; jumping ahead, a distribution  $\mathcal{T}$  for which there exists a guru that, in expectation, manages to retain followers to eternity (or until  $t = |\Gamma|$  for finite  $\Gamma$ ) will be said to be  *$r_0$ -retainable* and the *retention complexity* of  $\mathcal{T}$  will be the minimal  $r_0$  such that  $\mathcal{T}$  is  $r_0$ -retainable (see Definitions 7, 8).

When the user updates her retention parameter at the end of round  $t$ , she uses a function  $\mathcal{S}(\cdot, \cdot, \cdot)$  that is real-valued and takes three inputs: (i) the guru’s predicted distribution  $P_{\gamma_t}$ ; (ii) the follower’s assessment of that distribution  $Q_{\gamma_t}$ ; and (iii) the value  $u_{\gamma_t}$  that materialized, sampled by Nature. The retention parameter at time  $t$  is given by

$$r_t = r_{t-1} + \mathcal{S}(P_{\gamma_t}, Q_{\gamma_t}, u_{\gamma_t}) \tag{1}$$

► **Remark (Simplifying assumptions).** The formula (1) makes the following assumptions on the follower’s update rule: that it is Markovian, uses  $r_{t-1}$  additively and does not depend on the follower’s identity nor on the identity of the attribute  $\gamma_t$  being predicted. Such assumptions are common when modeling human behavior and we leave the study of more general update functions to future work.

### 1.3 Retentive Scoring Rules

The definition of the function  $\mathcal{S}$  above, and the surprising corollaries of this definition, are what dominates the first part of our study. We assume  $\mathcal{S}$  belongs to a class of functions that *elicit* the true beliefs of both guru and follower regarding the distribution for the attribute  $\gamma_t$ . Truth eliciting rules are ones that incentivize (rational) players to supply the rule with what they believe to be the truth. A famous early example of a truth eliciting rule is that of a one-party *proper scoring rule*, which will be tightly related to our two-party retentive scoring rule  $\mathcal{S}$ , so we start with the simpler, one-party, case.

#### Proper (one-party) Scoring Rules

One-party *proper scoring rules* are used to compensate a single forecaster of Nature in a truth-eliciting manner; these rules are studied extensively in the Decision Theory literature [17, 23, 11] and have interesting connections to the fields of estimation, information theory, and machine learning; see [8] for a recent survey. A scoring rule receives a single forecast, which is a distribution  $P$  over  $\mathcal{X}$  as an input (say, this could be the temperature at noon tomorrow at a fixed location), and scores the forecaster based on the outcome selected by Nature (the actual temperature). A scoring rule is called *proper* if it is maximized by forecasting the true distribution. We recite the definition as it appears in [23, 11]:

► **Definition 1** (Proper Scoring Rule). Let  $\mathcal{P}$  be a convex set of distributions over an arbitrary input space  $\mathcal{X}$ . A (one-party) *scoring rule* is a function  $s : \mathcal{P} \times \mathcal{X} \rightarrow \mathbb{R}$ . The scoring rule  $s$  is *proper* with respect to  $\mathcal{P}$  if, for all  $R \in \mathcal{P}$  (viewed as Nature's true distribution), the expected score  $\mathbb{E}_{x \sim R} [s(P, x)]$  is maximized over  $P \in \mathcal{P}$  at  $P = R$ :

$$\forall P \in \mathcal{P} \quad \mathbb{E}_{x \sim R} [s(P, x)] \leq \mathbb{E}_{x \sim R} [s(R, x)] \quad (2)$$

Intuitively, when the agent forecasts a distribution  $P \in \mathcal{P}$  and event  $x \in \mathcal{X}$  materializes, the reward for the expert is  $s(P, x)$ . To increase clarity when one-party and two-party (retentive) scoring rules are involved, we will use a lowercase  $s$  to denote a proper (one-party) scoring rule, and a calligraphic  $\mathcal{S}$  to denote a retentive (two-party) one.

Many proper scoring rules can be constructed using elementary functions, for example the *logarithmic scoring rule*:

$$s(P, i) = \log p_i \quad (3)$$

and *Brier's scoring rule* [6]:

$$s(P, i) = 2p_i - \sum_j p_j^2 = 2p_i - \|P\|_2^2 \quad (4)$$

#### Retentive (Two-Party) Scoring Rules

In the spirit of proper scoring rules, we define a *retentive* scoring rule which involves two parties: guru and follower. Using an axiomatic approach, we start by defining the desired properties of such a rule:

► **Definition 2** (Retentive Scoring Rule). Let  $\mathcal{P}$  be a convex set of distributions over an arbitrary input space  $\mathcal{X}$ . A function  $\mathcal{S} : \mathcal{P} \times \mathcal{P} \times \mathcal{X} \rightarrow \mathbb{R}$  is called a *retentive scoring rule* if it satisfies the following conditions:

1. *Cost of ignorance*: For all distributions  $P \in \mathcal{P}$  and outcomes  $x \in \mathcal{X}$ ,

$$\mathcal{S}(P, P, x) = -1 \tag{5}$$

2. *Proper scorings*: for any distribution  $R \in \mathcal{P}$  dictated by Nature:

- a. *Guru-side*: For any fixed follower belief  $Q \in \mathcal{P}$ , the best guru prediction  $P \in \mathcal{P}$  is Nature's:

$$\mathbb{E}_{x \sim R} [\mathcal{S}(P, Q, x)] \leq \mathbb{E}_{x \sim R} [\mathcal{S}(R, Q, x)] \tag{6}$$

- b. *Follower-side*: For any fixed guru prediction  $P \in \mathcal{P}$ , the best follower belief  $Q \in \mathcal{P}$  is Nature's:

$$\mathbb{E}_{x \sim R} [\mathcal{S}(P, Q, x)] \geq \mathbb{E}_{x \sim R} [\mathcal{S}(P, R, x)] \tag{7}$$

Intuitively, the *cost of ignorance* condition models the “attention economy” cost of interaction, and captures the intuition that the follower will penalize gurus that are no “smarter” than he is. For example, no guru/meteorologist will get followers by predicting “100% chance of sun in the Sahara desert”. The predictions must be surprising to the followers. In the formal definition, the penalty constant is normalized to  $-1$  to simplify analysis.

The output of  $\mathcal{S}$  is a quantity that the guru wishes to maximize, because doing so will mean the follower is retained longer, as seen by Equation (1). Therefore, the *guru-side properness* requirement (Equation (6)) implies that a rational guru will strive to report the correct distribution used by Nature ( $R$ ), if the guru knows that distribution. In other words, we require the scoring rule to elicit *truthful* guru-side inputs.

Similarly, since the follower has a limited attention span, she is incentivized to judge the guru's quality “honestly”, and this is modeled by the *follower-side properness* condition (Equation (7)); it means the follower too will supply the rule  $\mathcal{S}$  with Nature's distribution, if known to her. Notice that the combination of the cost-of-ignorance and two properness results mean that a rational guru will not offer “obvious advice” about which both guru and follower “know the (same) truth”.

### Retentive Rule Construction

One-party scoring rules give rise to two-party retentive scoring rules in a straightforward way: Score the guru and follower *separately* based on Nature's outcome using, perhaps, two different functions, and define the retentive score as the difference between the one-party scores minus a fixed constant (to account for the cost-of-ignorance (5)). A retentive scoring rule of this form is said to be *separable*, and a special case is that of a *symmetric* rule, in which both guru and follower are scored using the same (one-party) scoring rule, formally:

► **Definition 3** (Symmetric Retentive Scoring Rule). A retentive scoring rule  $\mathcal{S} : \mathcal{P} \times \mathcal{P} \times \mathcal{X} \rightarrow \mathbb{R}$  is called *symmetric* if there exists a proper one-party scoring rule  $s : \mathcal{P} \times \mathcal{X} \rightarrow \mathbb{R}$  such that:

$$\mathcal{S}(P, Q, i) = s(P, i) - s(Q, i) - 1 \tag{8}$$

### Characterization

Restricting the discussion to *categorical distributions*, i.e., to cases where  $\mathcal{X}$  is finite, and assuming the retentive scoring rules are *analytic*, meaning that a uniformly convergent power series expansion exists about any  $P \in \mathcal{P}$ , our first main result is the following statement:

## 12:6 The Complexity of User Retention

► **Theorem 4** (Retentive Scoring Rules are Symmetric). *The function  $\mathcal{S} : \mathcal{P} \times \mathcal{P} \times \mathcal{X} \rightarrow \mathbb{R}$  is an analytic retentive scoring rule for categorical distributions if and only if there exists a proper and analytic scoring rule  $s : \mathcal{P} \times \mathcal{X} \rightarrow \mathbb{R}$  such that:*

$$\mathcal{S}(P, Q, x) = s(P, x) - s(Q, x) - 1 \quad (9)$$

We find the statement somewhat surprising because it is not intuitively clear that a two-party retentive rule must be separable; symmetry follows rather directly from separability and the cost-of-ignorance assumption. For the proof – given in Section 2 – we use a known result which relates proper scoring rules to convex functions over the probability simplex. We show that each retentive scoring rule corresponds to a solution of a system of partial differential equations (PDEs). Solving the system and characterizing the family of solutions yields the result (see Subsection 2.2).

### 1.4 Memory Span

To model the different prediction capacities of gurus and followers, the forecasting abilities of both types of agents in the Collaborative Discovery model are characterized by a parameter called *memory span*, defined below.

A variety of psychological studies could be summarized by saying that the human short-term memory has a capacity of about “seven, plus-or-minus two” *chunks*, where each chunk can be roughly defined as a collection of elementary information relating to a single concept [18, 24]. What counts as a chunk depends on the knowledge of the person being tested. For instance, a word is a single chunk for a speaker of the language but is many chunks for someone who is totally unfamiliar with the language and sees the word as a collection of phonetic segments.

In the world of child development, young parents (who usually don’t have significant experience or formal child-development education) are likely to predict that their child will start walking around the average time for the entire population. Child development experts, on the other hand, usually have better ability to pick up subtle developmental signals from observed child behavior, and provide a better prediction based on them.

In this spirit, we proceed with the formal definition. In what follows, let  $\Delta(\mathcal{X}^\Gamma)$  denote the simplex of probability distributions over  $\mathcal{X}^\Gamma$  and  $\Delta(\mathcal{X})$  is the simplex of distributions over  $\mathcal{X}$ .

► **Definition 5** (Memory Span). Let  $\mathcal{T} \in \Delta(\mathcal{X}^\Gamma)$  be a distribution. An agent is said to have *memory span*  $m \geq 0$  when its prediction  $P_\gamma \in \Delta(\mathcal{X})$  for coordinate  $\gamma_t \in \Gamma$  of an instance  $u \in \mathcal{X}^\Gamma$  with disclosed parameters  $\Gamma_t \subseteq \Gamma$  (i.e. for which  $u_{\Gamma_t}$  is known) is based on  $m$  disclosed coordinates or less:

$$\forall \gamma \in \Gamma, \exists I_t \subseteq \Gamma_t : |I_t| \leq m, P_\gamma = (\mathcal{T}_\gamma \mid u_{\setminus I_t}) \quad (10)$$

where  $(\mathcal{T}_\gamma \mid u_{\setminus I_t})$  is the marginal distribution of  $\mathcal{T}$  on coordinate  $\gamma$ , conditioned on the event that the coordinates  $I_t$  are set to  $u_{\setminus I_t}$ .

Intuitively, this means that every prediction of an agent is based on its entire knowledge of at most  $m$  coordinates. When  $m = 0$ , a prediction is only based on the marginal distribution of the corresponding parameter in the entire population.

### 1.4.1 Monotonicity

Our second result, stated next, says that if guru  $G$  is “smarter” than guru  $G'$ , meaning her memory span ( $m_g$ ) is greater than his ( $m'_g$ ), the smarter guru  $G$  will also have higher success in retaining followers, in expectation. (Whether this optimistic result holds in the real world is highly debatable.) This result is not implied directly by the definition of the Collaborative Discovery model, and shows that our model exhibits intuitive and desirable properties that substantiate its theoretical appeal:

► **Theorem 6** (Knowledgeable Gurus Retain Better). *Let  $\mathcal{S} : \mathcal{P} \times \mathcal{P} \times \mathcal{X} \rightarrow \mathbb{R}$  be an analytic retentive scoring rule, let  $G_1, G_2$  be two gurus with memory spans  $m_g^{(1)} \geq m_g^{(2)}$ . Then for any distribution  $\mathcal{T}$ , any coordinate  $x$ , and follower with memory span  $m_f \leq m_g^{(2)}$ :*

$$\mathbb{E}_{\mathcal{T}}[\mathcal{S}(P_1, Q, x)] \geq \mathbb{E}_{\mathcal{T}}[\mathcal{S}(P_2, Q, x)] \quad (11)$$

where  $P_1, P_2 \in \Delta(\mathcal{X})$  are the distributional forecasts of gurus  $G_1$  and  $G_2$  respectively, and  $Q \in \Delta(\mathcal{X})$  is the belief of the follower.

A technical discussion of the theorem and its proof are provided in Section 3.

### 1.4.2 Retainability as a Function of Memory Span Discrepancy

From here on we assume that the guru has memory span  $m_g$ , and her follower has memory span  $m_f$  and moreover, both parties provide to the retentive scoring rule a distribution that is the correct marginal  $\mathcal{T}_{\gamma_t} \mid u_{\setminus J_t}$ , conditioned on some subset of  $J_t \subset \Gamma_t$  of size  $m_g$  for the guru and  $m_f$  for the follower, respectively. Under this assumption, notice that if  $m_g = m_f$  then both parties supply the same distribution, so the cost-of-ignorance assumption of Definition 2 means the follower will terminate the interaction within  $r_0$  steps; in other words, ignorant gurus will not prevail. Henceforth assume  $m_g > m_f$ . Combining the concepts of limited user attention, retentive scoring, and limited memory span, we can now ask: Is it possible for the guru to retain her follower throughout the process? This leads to the concept of *retainability*:

► **Definition 7** (Retentively Learnable Distribution). *Let  $\mathcal{T} \in \Delta(\mathcal{X}^{\Gamma})$ , and assume  $|\Gamma| = n$ . Given a retentive scoring rule  $\mathcal{S} : \mathcal{P} \times \mathcal{P} \times \mathcal{X} \rightarrow \mathbb{R}$ , guru memory span  $m_g \geq 0$ , follower memory span  $m_f \geq 0$ , and an initial retention parameter  $r_0 > 0$ , we say that  $\mathcal{T}$  is *retentively learnable* with respect to  $(\mathcal{S}, m_g, m_f, r_0)$  if there exists an ordering  $\gamma_1, \dots, \gamma_n$  of  $\Gamma$ , and a sequence of sets  $I_1, \dots, I_n$  such for all  $t \in [n]$ :*

1.  $I_t \subseteq \{\gamma_1, \dots, \gamma_{t-1}\}$
2.  $|I_t| \leq m_g$
3. For every sequence of sets  $J_1, \dots, J_n$  such that  $J_t \subseteq \{\gamma_1, \dots, \gamma_{t-1}\}$ ,  $|J_t| \leq m_f$ :

$$r_t = r_0 + \sum_{t'=1}^t \mathcal{S}((\mathcal{T}_{\gamma_{t'}} \mid u_{\setminus I_{t'}}), (\mathcal{T}_{\gamma_{t'}} \mid u_{\setminus J_{t'}}), \mathcal{T}) \geq 0 \quad (12)$$

Intuitively, a probability distribution is retentively learnable when it is possible to maintain a positive retention parameter throughout the process. From (12) we can see that increasing  $r_0$  does not hurt retainability. In other words, for  $r'_0 > r_0$ , if a distribution is retentively learnable with respect to  $(\mathcal{S}, m_g, m_f, r_0)$ , then it is also retentively learnable for  $(\mathcal{S}, m_g, m_f, r'_0)$ . We know that attention is a very limited resource, so we cannot expect it to be arbitrarily large. This leads to the following question: How large should the “initial retention” be in order for the guru to sustain her follower throughout the collaborative discovery process?

► **Definition 8** (Retention Complexity). The *retention complexity* of a distribution  $\mathcal{T} \in \Delta(\mathcal{X}^\Gamma)$  with respect to  $(\mathcal{S}, m_g, m_f)$  is the minimal value of  $r_0$  such that  $\mathcal{T}$  is retainable:

$$r_{\mathcal{S}, m_g, m_f}(\mathcal{T}) = \min \{r_0 \mid \mathcal{T} \text{ is retainable with respect to } (\mathcal{S}, m_g, m_f, r_0)\} \quad (13)$$

The discussion above leads to the following notion of retentively learnable families of distributions and properties. Recall that a property  $P$  over alphabet  $\mathcal{X}$  is a set of finite strings over  $\mathcal{X}$ . Let  $P_n = P \cap \mathcal{X}^n$  denote the set of strings in  $P$  of length  $n$ . Let  $\mathbb{N}_P = \{n \in \mathbb{N} \mid P_n \neq \emptyset\}$  be the set of lengths of strings that belong to  $P$ . The *family of distributions induced by  $P$*  is the family of uniform distributions over  $P_n$ , defined for  $n \in \mathbb{N}_P$ .

► **Definition 9** (Retentively Learnable Distributions and Properties). Let  $\mathbb{D} = \{\mathcal{T}_i\}_{i \in I}$  be a family of categorical distributions, where each  $\mathcal{T}_i$  is supported on strings of length  $n_i$  over alphabet  $\mathcal{X}$  (the set  $I$  may be infinite). Let  $\mathcal{S} : \mathcal{P} \times \mathcal{P} \times \mathcal{X} \rightarrow \mathbb{R}$  be a scoring rule as in Definition 3. We say  $\mathbb{D}$  is *retentively learnable* with respect to  $\mathcal{S}$  if there exist constants  $r_0$  and  $m_g > m_f$  such that each  $\mathcal{T}_i \in \mathbb{D}$  is retainable with respect to  $(\mathcal{S}, m_g, m_f, r_0)$ .

Similarly, a property  $P \subset \mathcal{X}^*$  is said to be *retentively learnable* with respect to  $\mathcal{S}$  if the family of distributions induced by  $P$  is retentively learnable.

Our main complexity result is the following. Recall that the property of linear functions over  $\mathbb{F}_2$  is the set of functions  $f : \mathbb{F}_2^k \rightarrow \mathbb{F}_2$  that satisfy  $f(x+y) = f(x) + f(y)$  for all  $x, y \in \mathbb{F}_2^k$ . For this property we have  $\mathbb{N}_P = \{n = 2^k \mid k \in \mathbb{N}\}$ .

► **Theorem 10** (Linear functions are retentively learnable). *The property of linear functions over the two-element field  $\mathbb{F}_2$  is retentively learnable.*

We elaborate on this result, and related ones, next.

## 1.5 The Retention Complexity of Linear Spaces

To initiate the study of the retentive learning within the context of computational complexity, a natural starting point is that of uniform distributions over linear spaces; linearity is the first object of study in other notable fields of complexity, like property testing [5, 13] and PAC learning [25].

Consider a realization of the model in which each attribute ranges over a binary space, i.e.,  $\mathcal{X}^\Gamma = \{0, 1\}^n$ . The Binary Attributes model describes a universe where each attribute is either present or not for a given user.

We start by redefining the problem using finite-field linear algebra, and then study the retention complexity of several natural families of linear codes, including the Walsh-Hadamard codes and the family of random Low Density Parity Check (LDPC) codes.

In particular, identify  $\{0, 1\}$  with the two-element finite field  $\mathbb{F}_2$  and consider a uniform distribution  $\mathcal{U}$  over a linear space  $U \subseteq \mathbb{F}_2^n$ . Let  $U^\perp$  denote the space dual to  $U$ . Let  $d(U)$  denote the Hamming distance of  $U$  (and  $d(U^\perp)$  is its dual distance), recalling that distance is equal to the minimum Hamming weight of a non-zero word in  $U$  (or  $U^\perp$ , respectively). We assume the guru has infinite memory span and the follower has memory span 0. (The study of the general case of  $0 < m_f < m_g < \infty$  is left for future work.) This means the follower's distribution for each  $i \in [n]$  is the uniform distribution on  $\mathbb{F}_2$  (this assumes  $U$  is not constant on any  $i \in [n]$ ).

We shall use a retentive scoring rule denoted  $\mathcal{S}_{\text{bin}}$ , that has expected value 1 when the guru can predict the next coordinate exactly, i.e., when the value of that coordinate depends linearly on the values of coordinates exposed thus far, and has expected value  $-1$  otherwise, when the distribution on that coordinate is linearly independent of all previously revealed bits.

The following result sets the bounds for our study of retention complexity in this setting, establishing a connection between the retention complexity of  $\mathcal{U}$  and its algebraic properties:

► **Lemma 11** (Retention Complexity Bounds for Linear Spaces). *For a uniform distribution  $\mathcal{U}$  over a linear space  $U \subseteq \mathbb{F}_n^2$  with unbounded guru memory span and zero follower memory span, the retention complexity satisfies:*

$$d(U^\perp) - 1 \leq r_{(\mathcal{S}_{\text{bin}}, \infty, 0)}(\mathcal{U}) \leq \dim(U) \quad (14)$$

Next, we show that the both bounds are tight. We begin by showing that a uniform distribution over codewords of the *Walsh-Hadamard* (WH) code achieves the lower retention complexity bound:

► **Lemma 12** (Walsh-Hadamard Retention Complexity). *For all  $k \in \mathbb{N}$ , a  $k$ -dimensional Walsh-Hadamard code satisfies:*

$$r_{(\mathcal{S}_{\text{bin}}, 2, 0)}(\text{WH}) = 2 \quad (15)$$

As the  $n$ -dimensional Walsh-Hadamard code represents the set of linear functions over  $\mathbb{F}_n^2$ , proving this lemma will also imply Theorem 10.

Finally, we show that a random LDPC code achieves the upper bound (up to multiplicative constants) with high probability, implying that collaborative discovery can be very hard on arbitrary linear spaces:

► **Theorem 13** (LDPC Retention Complexity). *For a proper choice of constants  $c, d > 0$  and sufficiently large  $n$ , the retention complexity of a random  $(c, d)$ -regular LDPC code over  $\mathbb{F}_2^n$  is linear with high probability:*

$$r_{(\mathcal{S}_{\text{bin}}, \infty, 0)}(\text{LDPC}) \stackrel{\text{w.h.p.}}{=} \Omega(\dim(\text{LDPC})) \quad (16)$$

The proofs of these results are provided in Subsection 4.2. The most technically challenging one is the third one and relies on the lower bounds for the testability of random LDPC codes of [4].

## 1.6 Related work

The study of reputation systems is interested in ranking gurus in “meaningful” ways, and is highly investigated empirically and theoretically; cf. [20, 21] and references therein. Closest in rigour to our approach are the papers by (i) Ban and Linial [2] which uses the theory of random processes to identify situations where gurus (called “experts” there) can be robustly ranked, assuming user participation continues indefinitely, and (ii) Chan et al. [7] that classifies interactive crowd-computation games using a small list of modeling parameters.

In the context of machine learning, the task of detecting users who are likely to stop participating in a voluntary system is known as *churn prediction*. For this task, machine learning algorithms are trained to recognize typical usage patterns and predict the likelihood of a termination [26, 9]. Even though general machine learning models provide good “black-box” churn predictors when trained correctly, gaining deep understanding of the underlying phenomena might be challenging.

Comparing our model to prior work, there are two main differences. First, our aim is to model the dynamics of *long-term* interaction between a follower and her guru about a *single complex phenomenon* of interest, asking when do followers abandon their gurus. Second, we



are interested in the *mathematical properties of phenomena* that are prone to collaborative discovery, meaning that for these phenomena a “good” guru will successfully instruct her followers from start to end without losing their attention and faith. This motivation is somewhat similar to that taken in the field of Property Testing [13] which attempts to understand which properties are amenable to “testing”.

## 1.7 Discussion of main contributions and future directions

The properties of retentive scoring rules, the effect of memory span discrepancy on the retention of followers, and the study of retention complexity of specific distributions are the main topics of this work.

We point out a few questions that emerge from the paper:

1. The gurus and followers studied here are assumed to have optimal knowledge of the distribution, up to their memory span limit. In particular, a guru with infinite memory span does not need to *learn* the distribution at all. However, in most realistic settings the distribution is unknown, leading to the question of learning distributions in a way that also maintains good retention properties. For instance, suppose the distribution is an *unknown* linear space  $U$  with retention complexity  $r$ . What is the minimal number of followers with initial retention parameter  $r_0 > r$  (say,  $r_0 = 2 \cdot r$ ) that will be “spent” or “lost” by the guru before she learns enough about  $U$  to fully retain new followers? This particular question is highly relevant to automated gurus that seek to attract users while maintaining high reputation (e.g., high app-store ratings).
2. The gurus and followers used here are computationally unbounded (or, more precisely, bounded only by attention span). Realistically, the computational complexity of computing marginals and evaluating which new attribute  $\gamma_t$  to interact about will be highly non-trivial.
3. Walsh-Hadamard codes are locally testable, correctable and decodable, while random LDPC codes have none of these properties; moreover, the retention complexity for both families of codes is approximately equal to their query complexity (for testability and correctability). This leads to our first question: *Are there tighter connections between retention complexity and query complexity of locally testable/correctable codes?* Do all  $q$ -query locally testable (or correctable) codes have retention complexity  $f(q)$  for some function that depends only on  $q$  and is independent of  $n$  (input size)? Likewise, it seems interesting to ask whether retention complexity is related to basic machine learning measures like VC dimension.

## 2 Retentive Scoring

In this section we study retentive scoring rules, and prove Theorem 4.

### 2.1 Preliminaries and Notations

#### Categorical Probability Distributions

Recall that a *categorical distribution* is a discrete probability distribution that describes the possible results of a random event that can take one of  $K$  possible outcomes. In this section, we assume  $\mathcal{P}$  is a convex set of categorical distributions with  $K = (n + 1)$  possible outcomes, i.e.  $\mathcal{X} = \{0, \dots, n\}$ . We define the number of possible outcomes as  $n + 1$  instead of  $n$  to simplify later calculations.



In addition, recall that the space of categorical distributions with  $(n + 1)$  possible outcomes is equivalent to the  $n$ -dimensional simplex:

$$\mathcal{P} \subseteq \Delta^n = \left\{ (p_0, \dots, p_n) \in \mathbb{R}^{n+1} \mid \sum_i p_i = 1; \forall i : p_i \geq 0 \right\} \quad (17)$$

where  $p_i$  is the probability of categorical event  $i$ .

### Expected Score Notation

Recall Definition 2. Following the conventions of the proper scoring literature, and given probability distributions  $P, Q, R \in \mathcal{P}$ , we denote the *expected retentive score* as:

$$\mathcal{S}(P, Q, R) \equiv \mathbb{E}_{i \sim R} [\mathcal{S}(P, Q, i)] \quad (18)$$

To avoid difficulties in (18), we will assume  $\mathcal{S}(P, Q, R)$  exists and is finite. Similarly, for one-party scoring rules, the common notation of *expected score* is:

$$s(P, R) \equiv \mathbb{E}_{i \sim R} [s(P, i)] \quad (19)$$

The analysis below will use both the single event notation  $\mathcal{S}(P, Q, i)$  and the expected score notation  $\mathcal{S}(P, Q, R)$  (and similarly for one-party scoring rules). To avoid confusion, we will always use upper-case letters to denote random variables and lower-case letters to denote events.

► **Remark (Scoring Rules on Infinite Sample Spaces).** Similar to proper (one-party) scoring rules, it is possible to define retentive scoring rules on infinite sample spaces using measure-theoretic tools. Computers are finite, and therefore many applications can be modeled as finite-dimensional categorical distributions. In this work we consider the finite sample space for concreteness and simplicity, and leave the rigorous measure-theoretic analysis to future work.

### Characterization of Proper Scoring Rules

One of the fundamental results in the research of proper scoring rules is the characterization theorem, first stated by [17], which defines a correspondence between proper scoring rules and convex functions over the probability simplex. We start with some preliminary definitions, and proceed with the characterization theorem itself:

► **Definition 14 (Subgradient).** A function  $\nabla^*G : \mathcal{P} \rightarrow \mathbb{R}^{n+1}$  is a subgradient of  $G$  at the point  $P$  if the following inequality holds for all  $Q \in \mathcal{P}$ :

$$G(Q) \geq G(P) + \langle \nabla^*G(P), (Q - P) \rangle \quad (20)$$

where  $\langle \cdot, \cdot \rangle$  denotes the euclidean inner product over  $\mathbb{R}^{n+1}$ :  $\langle X, Y \rangle = \sum_{i=0}^n x_i y_i$ .

► **Remark (Subgradients of Differentiable Functions).** If  $G$  is differentiable at  $P \in \mathcal{P}$  then  $G$  has a *unique* subgradient at  $P$  and it equals the gradient  $\nabla G = \left( \frac{\partial G}{\partial p_0}, \dots, \frac{\partial G}{\partial p_n} \right)$  at  $P$ .

Recall that a real-valued function  $G : \mathcal{P} \rightarrow \mathbb{R}$  is convex if:  $G((1 - \lambda)P + \lambda Q) \leq (1 - \lambda)G(P) + \lambda G(Q)$  for all  $P, Q \in \mathcal{P}$  and  $\lambda \in [0, 1]$ .

► **Lemma 15** ([15], Theorem 2.1).  $G : \mathcal{P} \rightarrow \mathbb{R}$  is convex if and only if it has a subgradient  $\nabla^*G$  at each point  $P \in \mathcal{P}$ .

► **Theorem 16** (McCarthy's Theorem, [11]). *A scoring rule  $s : \mathcal{P} \times \Omega \rightarrow \mathbb{R}$  is proper relative to  $\mathcal{P}$  if and only if there exists a convex, real-valued function  $G$  on  $\mathcal{P}$  such that:*

$$s(P, i) = G(P) - \langle \nabla^* G(P), P \rangle - (\nabla^* G)_i \quad (21)$$

where  $(\nabla^* G)_i$  is the  $i$ th component of  $(\nabla^* G)$ .

We also define the *Generalized Entropy* as the convex function which is induced by the proper scoring rule:

► **Definition 17** (Generalized Entropy). The convex function  $G(P) = s(P, P)$  induced by a proper scoring rule  $s$  is called the *generalized entropy function* of  $s$ .

Note that a convex general entropy function exists for every proper scoring rule by Theorem 16. For the logarithmic scoring rule defined in (3), the associated general entropy function is the additive inverse of the Shannon entropy:  $G(P) = \sum_{i=0}^n p_i \log p_i$ . Additional information-theoretic quantities can be generalized using proper scoring rules. See [8] for a recent review.

## 2.2 Separability of Retentive Scoring Rules

In this section, we prove that every proper retentive scoring rule can be written as the difference between two proper scoring rules. Recall Theorem 4:

► **Theorem 4** (Retentive Scoring Rules are Symmetric). *The function  $\mathcal{S} : \mathcal{P} \times \mathcal{P} \times \mathcal{X} \rightarrow \mathbb{R}$  is an analytic retentive scoring rule for categorical distributions if and only if there exists a proper and analytic scoring rule  $s : \mathcal{P} \times \mathcal{X} \rightarrow \mathbb{R}$  such that:*

$$\mathcal{S}(P, Q, x) = s(P, x) - s(Q, x) - 1 \quad (9)$$

The proof has several steps:

1. We verify that symmetric retentive scoring rules are indeed proper (Lemma 18).
2. Conversely, we first define the notion of a *separable scoring rule*, which is a rule which can be written as the difference between two one-party scoring rules. Given a retentive scoring rule, we use the proper scoring characterization theorem (Theorem 16) to construct a system of partial differential equations which describes the constraints that must be satisfied by such a rule (Lemma 20). We then solve the characterizing system of partial differential equations (Lemma 21), and show that every possible solution corresponds to a separable retentive scoring rule (Lemma 22).
3. Finally, we show that every separable retentive scoring rule with constant cost of ignorance is also symmetric, proving the theorem.

We proceed by stating and proving the lemmas, and conclude the section by proving the theorem itself.

### Preliminaries

The proofs of Lemma 18 and Lemma 20 rely on the formalism of proper scoring rules and retentive scoring rules. The proof of Claim 21 relies on basic results from the theory of quasi-linear partial differential equations (refer to [19] for a thorough introduction). For  $D \subseteq \mathbb{R}^n$ , we will refer to a function  $f : D \rightarrow \mathbb{R}$  as *analytic* if its Taylor expansion about  $\mathbf{x} \in D$  converges to  $f(x)$  for all  $\mathbf{x} \in D$ . We use  $e_i \in \mathbb{R}^n$  to denote the  $i$ th vector of the standard basis. The gradient of a differentiable function  $g(\mathbf{x}, \mathbf{y}) : \mathbb{R}^n \times \mathbb{R}^n \rightarrow \mathbb{R}$  with respect to  $\mathbf{x} \in \mathbb{R}^n$  is denoted by  $\frac{\partial g}{\partial \mathbf{x}} \equiv \left( \frac{\partial g}{\partial x_1}, \dots, \frac{\partial g}{\partial x_n} \right)^T$ .

### 2.2.1 Symmetric Rules are Retentive

► **Lemma 18.** *Let  $\mathcal{S} : \mathcal{P} \times \mathcal{P} \times \mathcal{X} \rightarrow \mathbb{R}$  be a scoring rule. If there is a proper scoring rule  $s : \mathcal{P} \times \mathcal{X} \rightarrow \mathbb{R}$  such that:  $\mathcal{S}(P, Q, i) = s(P, i) - s(Q, i) - 1$ , then  $\mathcal{S}(P, Q, i)$  is retentive.*

**Proof.** Let  $P, Q, R \in \mathcal{P}$ . Using (8), the expected score  $\mathcal{S}(P, Q, R)$  is:

$$\mathcal{S}(P, Q, R) = s(P, R) - s(Q, R) - 1 \quad (22)$$

$s$  is proper, and therefore  $s(P, R) \leq s(R, R)$ . Plugging into (22) we obtain:

$$\mathcal{S}(P, Q, R) \leq s_1(R, R) - s_2(Q, R) = \mathcal{S}(R, Q, R) \quad (23)$$

satisfying (6). Similarly,  $s_2$  is also proper, and therefore:

$$\mathcal{S}(P, Q, R) \geq s_1(P, R) - s_2(R, R) = \mathcal{S}(P, R, R) \quad (24)$$

satisfying (6). For  $Q = P$  we get  $\mathcal{S}(P, P, i) = -1$  for all  $i \in \mathcal{X}$ , and therefore  $\mathcal{S}$  is retentive according to Definition 2. ◀

### 2.2.2 Retentive Rules are Separable

We start by formally defining the notion of a separable scoring rule:

► **Definition 19** (Separable Retentive Scoring Rule). A proper retentive scoring rule  $\mathcal{S} : \mathcal{P} \times \mathcal{P} \times \Omega \rightarrow \mathbb{R}$  is called *separable* if there exists two proper scoring rules  $s_1, s_2 : \mathcal{P} \times \Omega \rightarrow \mathbb{R}$  such that:

$$\mathcal{S}(P, Q, i) = s_1(P, i) - s_2(Q, i) \quad (25)$$

We also say that a two-party scoring rule is *proper* if it satisfies (6), (7). In the following lemma, we say that a bi-variate function  $G : \mathcal{P} \times \mathcal{P} \rightarrow \mathbb{R}$  is convex with respect to its first argument if  $G(P, Q)$  is a convex function of  $P$  for any constant  $Q \in \mathcal{P}$ ; convexity with respect to the second argument is similarly defined by switching the roles of  $P$  and  $Q$ .

► **Lemma 20** (Characterization by subgradients). *A two-party scoring rule  $\mathcal{S}$  is proper with respect to class  $\mathcal{P}$  if and only if there exist two functions  $G, H : \mathcal{P} \times \mathcal{P} \rightarrow \mathbb{R}$  such that:*

1.  $G(P, Q)$  is convex with respect to  $P$ .
2.  $H(P, Q)$  is convex with respect to  $Q$ .
3. For all  $P, Q, R \in \mathcal{P}$ :

$$G + \langle \nabla_P^* G, (R - P) \rangle = -(H + \langle \nabla_Q^* H, (R - Q) \rangle) \quad (26)$$

where  $\nabla_P^* G$  is a subgradient of  $G(P, Q)$  with respect to its first argument, and  $\nabla_Q^* H$  is a subgradient of  $H(P, Q)$  with respect to its second argument.

**Proof.** For the first direction, let  $\mathcal{S}(P, Q, i)$  be a proper retentive scoring rule, and define  $s_Q(P, i) \equiv \mathcal{S}(P, Q, i)$ . Using (6) we obtain that  $s_Q(P, R) \leq s_Q(R, R)$ . Therefore  $s_Q$  is proper, and according to Theorem 16 there exists a convex function  $G_Q : \mathcal{P} \rightarrow \mathbb{R}$  that depends on  $Q$ , such that:

$$s_Q(P, i) = G_Q(P) - \langle \nabla^* G_Q(P), P \rangle + (\nabla^* G_Q(P))_i \quad (27)$$

## 12:14 The Complexity of User Retention

where  $(\nabla^* G_Q(P))_i$  is  $i$ th entry of  $\nabla^* G_Q$  at point  $P$ . Similarly, define  $s_P(Q, i) \equiv \mathcal{S}(P, Q, i)$ . By the same reasoning and using (7) we obtain that  $-s_P$  is proper, and therefore there exists a convex function  $H_P : \mathcal{P} \rightarrow \mathbb{R}$  such that:

$$-s_P(Q, i) = H_P(Q) - \langle \nabla^* H_P(Q), Q \rangle + (\nabla^* H_P(Q))_i \quad (28)$$

Define  $G(P, Q) \equiv G_Q(P)$  and  $H(P, Q) \equiv H_P(Q)$ . Note that  $G$  is convex with respect to  $P$  and  $H$  is convex with respect to  $Q$ , satisfying conditions 1, 2. Let  $R \in \mathcal{P}$ . Using the fact that  $s_P(P, R) = s_Q(Q, R)$ , we can combine (27), (28) to obtain:

$$G + \langle \nabla_P^* G, (R - P) \rangle = -(H + \langle \nabla_Q^* H, (R - Q) \rangle) \quad (29)$$

satisfying condition 3.

Conversely, let  $G, H$  be the functions which satisfy the three conditions above. Define:

$$s_Q(P, i) \equiv G - \langle \nabla_P^* G, P \rangle + (\nabla_P^* G)_i \quad (30)$$

$$s_P(Q, i) \equiv -\left(H - \langle \nabla_Q^* H, H \rangle + (\nabla_Q^* H)_i\right) \quad (31)$$

Note that  $s_Q = -s_P$  by equation (26), and that  $s_P$  and  $-s_Q$  are proper by Theorem 16.

Define  $\mathcal{S}(P, Q, i) = s_Q(P, i) = -s_P(Q, i)$ .  $s_Q$  is proper, and therefore  $\mathcal{S}(P, Q, R) \leq \mathcal{S}(R, Q, R)$ , satisfying the properness condition in (6). Similarly, the properness of  $-s_P$  implies  $\mathcal{S}(P, R, R) \leq \mathcal{S}(P, Q, R)$ , satisfying (7), and therefore  $\mathcal{S}$  is proper. ◀

The following lemma contains a solution of a partial differential equation that will assist us in solving the characterizing equations of proper retentive scoring rules. We obtain the solution using basic tools from the theory of partial differential equations, and the proof is given in Appendix A for completeness:

► **Claim 21.** *Let  $D \subseteq \mathbb{R}^n$  such that  $\mathbf{x}, \mathbf{y} \in D$ . For every analytic function  $u : D \times D \rightarrow \mathbb{R}$  satisfying the equation*

$$u(\mathbf{x}, \mathbf{y}) - \sum_{i=1}^n (y_i - x_i) \frac{\partial u(\mathbf{x}, \mathbf{y})}{\partial x_i} = 0 \quad (32)$$

there exist functions  $\alpha_1, \dots, \alpha_n : D \rightarrow \mathbb{R}$  such that:

$$u(\mathbf{x}, \mathbf{y}) = \sum_{i=1}^n \alpha_i(\mathbf{y})(y_i - x_i) \quad (33)$$

The following lemma is the heart of this part of the proof of Theorem 4 .

► **Lemma 22** (Proper Retentive Rules are Separable). *Let  $\mathcal{S} : \mathcal{P} \times \mathcal{P} \times \mathcal{X} \rightarrow \mathbb{R}$  be a retentive scoring rule. If  $\mathcal{S}$  is a proper retentive scoring rule with analytic generalized entropy functions, then there exist two functions  $s_1, s_2 : \mathcal{P} \times \Omega \rightarrow \mathbb{R}$  such that  $\mathcal{S}(P, Q, i) = s_1(P, i) - s_2(Q, i)$ .*

Proof outline:

1. Given a proper retentive scoring rule, Lemma 20 implies the existence of two generalized entropy functions  $G, H : \mathcal{P} \times \mathcal{P} \rightarrow \mathbb{R}$  related by equation (26).
2. We choose a parametrization for points on the simplex  $\Delta^n$ , and use it to define (26) in the convex domain  $D = \{(x_1, \dots, x_n) \in \mathbb{R}_+^n \mid \sum_i x_i \leq 1\}$ .
3. We simplify the resulting equation, and solve it using Claim 21.

4. Applying the correspondence established in Theorem 16 between convex functions on the simplex and proper scoring rules, we show that the generalized entropy functions  $G, H$  induce a separable scoring rule.

Following the conventions of multivariate calculus, in the proof we will use the  $\cdot$  symbol to denote the euclidean inner product over  $\mathbb{R}^n$ :  $\mathbf{x} \cdot \mathbf{y} = \sum_{i=1}^n x_i y_i$ . In addition, the proof employs terms from the theory of multivariate convex analysis: Given a non-empty convex subset  $S \subseteq \mathbb{R}^n$ , its *affine hull*  $\text{Aff}(S)$  is the smallest affine set containing  $S$ . A *relative interior point* is a member of the set  $\{x \in S : \exists \epsilon > 0, N_\epsilon(x) \cap \text{Aff}(S) \subseteq S\}$ , where  $N_\epsilon(x)$  is the  $\epsilon$ -ball around point  $x$ . Refer to [22] for an introduction to convex analysis. In the proof, we also use the *gradient theorem* for line integrals, which is a common generalization of the fundamental theorem of calculus. We recall it here without proof. Refer to Wikipedia entry [14] for discussion and proof:

► **Claim 23** (Gradient Theorem). *Let  $\varphi : U \subseteq \mathbb{R}^n \rightarrow \mathbb{R}$  and  $\gamma$  is any curve from  $\mathbf{p}$  to  $\mathbf{q}$ . Then:*

$$\varphi(\mathbf{q}) - \varphi(\mathbf{p}) = \int_{\gamma[\mathbf{p}, \mathbf{q}]} \nabla \varphi(\mathbf{r}) \cdot d\mathbf{r} \quad (34)$$

**Proof of Lemma 22.** Let  $\mathcal{S}$  be a proper retentive scoring rule. By Lemma 20, there exist two functions  $G, H : \mathcal{P} \times \mathcal{P} \rightarrow \mathbb{R}$  such that  $G(P, Q)$  is convex with respect to its first argument,  $H(P, Q)$  is convex with respect to its second argument, and equation (26) is satisfied.

When  $P, Q$  and  $R$  are categorical random variables with  $n+1$  possible outcomes, equation (26) is defined over the  $n$ -dimensional simplex  $\Delta^n$ . Let  $D = \{(x_1, \dots, x_n) \in \mathbb{R}_+^n \mid \sum_i x_i \leq 1\}$ . Each point  $P$  on the simplex can be represented by a vector  $P = (p_0, \dots, p_n) \in \mathbb{R}_+^{n+1}$  such that  $\sum_{i=0}^n p_i = 1$ . To simplify the constraints, we define a bijection  $f : \Delta^n \rightarrow D$  as follows:

$$f(P) \equiv (p_1, \dots, p_n) \in \mathbb{R}^n \quad (35)$$

$$f^{-1}(\mathbf{x}) \equiv \left(1 - \sum_{i=1}^n x_i, x_1, \dots, x_n\right) \in \Delta^n \quad (36)$$

using this bijection, we represent each point on the simplex using a  $n$ -dimensional vector in the domain. Denote:  $P \equiv f^{-1}(\mathbf{x})$ ,  $Q \equiv f^{-1}(\mathbf{y})$ ,  $R \equiv f^{-1}(\mathbf{z})$ ,  $f(\mathcal{P}) \equiv \{f(P) \mid P \in \mathcal{P}\}$ .

Using this correspondence, we also define  $g(\mathbf{x}, \mathbf{y}) \equiv G(P, Q)$ ,  $h(\mathbf{x}, \mathbf{y}) \equiv H(P, Q)$ . The assumption that  $G, H$  are analytic implies that the gradients of each function coincide with their corresponding subgradients (See Remark 2.1).

We will now write (26) using the new parametrization. Let  $\mathbf{x}, \mathbf{y}, \mathbf{z} \in f(\mathcal{P})$ . For the left-hand side of (26) we obtain:

$$\frac{\partial g}{\partial \mathbf{x}} \cdot (\mathbf{z} - \mathbf{x}) = \sum_{i=1}^n \frac{\partial g}{\partial x_i} (z_i - x_i) \quad (37)$$

[Calculate the derivative of  $g$  using the chain rule]

$$= \sum_{i=1}^n \left( \frac{\partial G}{\partial p_i} - \frac{\partial G}{\partial p_0} \right) (z_i - x_i) \quad (38)$$

[Rearrange the summations]

$$= \frac{\partial G}{\partial p_0} \sum_{i=1}^n (-z_i + x_i) + \sum_{i=1}^n \frac{\partial G}{\partial p_i} \cdot (z_i - x_i) \quad (39)$$

## 12:16 The Complexity of User Retention

$$= \frac{\partial G}{\partial p_0} \left( \left( 1 - \sum_{i=1}^n z_i \right) - \left( 1 - \sum_{i=1}^n x_i \right) \right) + \sum_{i=1}^n \frac{\partial G}{\partial p_i} \cdot (z_i - x_i) \quad (40)$$

[Use the definition of  $\mathbf{x}, \mathbf{z}$ ]

$$= \sum_{i=0}^n \frac{\partial G}{\partial p_i} (r_i - p_i) \quad (41)$$

$$= \nabla G \cdot (R - P) \quad (42)$$

A similar argument on the right-hand side of (26) shows that  $\nabla H \cdot (R - Q) = h + \frac{\partial h}{\partial \mathbf{y}} \cdot (\mathbf{z} - \mathbf{y})$ , and therefore the system defined in (26) is equivalent to:

$$\forall \mathbf{x}, \mathbf{y}, \mathbf{z} \in f(\mathcal{P}) : g + \frac{\partial g}{\partial \mathbf{x}} \cdot (\mathbf{z} - \mathbf{x}) = - \left( h + \frac{\partial h}{\partial \mathbf{y}} \cdot (\mathbf{z} - \mathbf{y}) \right) \quad (43)$$

We will now simplify (43) using its linear properties. Denote the affine hull of  $f(\mathcal{P})$  by  $\text{Aff}(f(\mathcal{P})) \equiv \mathbf{v}_0 + V$ , and assume  $\mathbf{v}_0$  is a relative interior point. Taking  $\mathbf{z} = \mathbf{v}_0$  in equation (43) yields:

$$g + \frac{\partial g}{\partial \mathbf{x}} \cdot (\mathbf{v}_0 - \mathbf{x}) = - \left( h + \frac{\partial h}{\partial \mathbf{y}} \cdot (\mathbf{v}_0 - \mathbf{y}) \right) \quad (44)$$

Similarly, denote the  $i$ th basis vector of  $V$  by  $\bar{\mathbf{v}}_i$ . For any  $i \in [\dim V]$ , taking  $\mathbf{z} = \mathbf{v}_0 + \bar{\mathbf{v}}_i$  in equation (43), with appropriate scaling of  $\bar{\mathbf{v}}_i$  such that  $\mathbf{z} \in f(\mathcal{P})$ , yields:

$$\forall i \in [\dim V] : g + \frac{\partial g}{\partial \mathbf{x}} \cdot (\mathbf{v}_0 + \bar{\mathbf{v}}_i - \mathbf{x}) = - \left( h + \frac{\partial h}{\partial \mathbf{y}} \cdot (\mathbf{v}_0 + \bar{\mathbf{v}}_i - \mathbf{y}) \right) \quad (45)$$

Subtracting (44) from (45) we obtain:

$$\forall i \in [\dim V] : \frac{\partial g(\mathbf{x}, \mathbf{y})}{\partial \mathbf{x}} \cdot \bar{\mathbf{v}}_i = - \frac{\partial h(\mathbf{x}, \mathbf{y})}{\partial \mathbf{y}} \cdot \bar{\mathbf{v}}_i \quad (46)$$

thus  $\frac{\partial g(\mathbf{x}, \mathbf{y})}{\partial \mathbf{x}}$  and  $-\frac{\partial h(\mathbf{x}, \mathbf{y})}{\partial \mathbf{y}}$  are equal component-wise, and therefore  $\frac{\partial g(\mathbf{x}, \mathbf{y})}{\partial \mathbf{x}} + \frac{\partial h(\mathbf{x}, \mathbf{y})}{\partial \mathbf{y}}$  is orthogonal to the affine hull:

$$\forall \mathbf{v} \in V : \left( \frac{\partial g(\mathbf{x}, \mathbf{y})}{\partial \mathbf{x}} + \frac{\partial h(\mathbf{x}, \mathbf{y})}{\partial \mathbf{y}} \right) \cdot \mathbf{v} = 0 \quad (47)$$

Note that  $(\mathbf{z} - \mathbf{x}), (\mathbf{z} - \mathbf{y}), (\mathbf{y} - \mathbf{x}) \in V$ . Substitute (47) back into (43) to obtain:

$$g + \frac{\partial g}{\partial \mathbf{x}} \cdot (\mathbf{y} - \mathbf{x}) = -h \quad (48)$$

Apply  $\frac{\partial}{\partial \mathbf{y}}$  on both sides to get:

$$\frac{\partial g}{\partial \mathbf{y}} + \frac{\partial}{\partial \mathbf{y}} \sum_{i=1}^n \frac{\partial g}{\partial x_i} (y_i - x_i) = - \frac{\partial h}{\partial \mathbf{y}} \quad (49)$$

And using (47) again we obtain:

$$\frac{\partial}{\partial \mathbf{y}} \sum_{i=1}^n \frac{\partial g}{\partial x_i} (y_i - x_i) = 0 \quad (50)$$

Which is equivalent to:

$$\forall k \in [n] : \frac{\partial g}{\partial y_k} + \sum_{i=1}^n (y_i - x_i) \frac{\partial}{\partial x_i} \frac{\partial g}{\partial y_k} = 0 \quad (51)$$

This is a system of  $n$  independent first-order partial differential equations for each element in  $\frac{\partial g}{\partial \mathbf{y}}$ . Using Claim 21, we obtain the general solution for each  $k$ :

$$\forall k \in [n], \exists \alpha_{k,1}, \dots, \alpha_{k,n} : \frac{\partial g}{\partial y_k} = \sum_{i=1}^n \alpha_{k,i}(\mathbf{y})(y_i - x_i) \quad (52)$$

Packing back the equations to vector form, we define a matrix operator  $A : D \rightarrow \mathbb{R}^{n \times n}$  such that  $A_{i,j}[\mathbf{y}] = \alpha_{k,i}(\mathbf{y})$ . The system in (52) in now be compactly represented using matrix multiplication:

$$\frac{\partial g}{\partial \mathbf{y}} = A[\mathbf{y}] (\mathbf{y} - \mathbf{x}) \quad (53)$$

We now use the correspondence established in Theorem 16 to show that the generalized entropy functions  $G, H$  induce a separable scoring rule. Applying the gradient theorem (34) along the curve  $\gamma(t) = \mathbf{0} + t\mathbf{y}$  for  $t \in [0, 1]$  yields:

$$g(\mathbf{x}, \mathbf{y}) - g(\mathbf{x}, \mathbf{0}) = \int_0^1 \left( \mathbf{y}^T \left( \frac{\partial g}{\partial \mathbf{y}} \Big|_{\mathbf{x}, t\mathbf{y}} \right) \right) dt \quad (54)$$

$$= \int_0^1 (\mathbf{y}^T A[t\mathbf{y}] (t\mathbf{y} - \mathbf{x})) dt \quad (55)$$

Denote  $\psi(\mathbf{x}) \equiv g(\mathbf{x}, \mathbf{0})$  and  $\varphi(\mathbf{x}, \mathbf{y}) \equiv \int_0^1 (\mathbf{y}^T A[t\mathbf{y}] (t\mathbf{y} - \mathbf{x})) dt$ . Note that  $\varphi(\mathbf{x}, \mathbf{y})$  is a linear function of  $\mathbf{x}$ . The scoring rule which corresponds to  $g$  is given by Theorem 16:

$$S(\mathbf{x}, \mathbf{y}, \mathbf{z}) = g(\mathbf{x}, \mathbf{y}) + \frac{\partial g(\mathbf{x}, \mathbf{y})}{\partial \mathbf{x}} \cdot (\mathbf{z} - \mathbf{x}) \quad (56)$$

$$= \underbrace{\psi(\mathbf{x}) + \frac{\partial \psi(\mathbf{x})}{\partial \mathbf{x}} \cdot (\mathbf{z} - \mathbf{x})}_{\equiv s_1} + \underbrace{\varphi(\mathbf{x}, \mathbf{y}) + \frac{\partial \varphi(\mathbf{x}, \mathbf{y})}{\partial \mathbf{x}} \cdot (\mathbf{z} - \mathbf{x})}_{\equiv s_2} \quad (57)$$

The terms denoted by  $s_1$  only depend on  $\mathbf{x}$  and  $\mathbf{z}$ , and therefore  $s_1 = s_1(\mathbf{x}, \mathbf{z})$ . In addition,  $\varphi(\mathbf{x}, \mathbf{y})$  is a linear function of  $\mathbf{x}$  and therefore both  $\frac{\partial \varphi(\mathbf{x}, \mathbf{y})}{\partial \mathbf{x}}$  and  $\left( \varphi(\mathbf{x}, \mathbf{y}) - \frac{\partial \varphi(\mathbf{x}, \mathbf{y})}{\partial \mathbf{x}} \cdot \mathbf{x} \right)$  do not depend on  $\mathbf{x}$ , thus  $s_2 = s_2(\mathbf{y}, \mathbf{z})$ . The scoring rule  $S(\mathbf{x}, \mathbf{y}, \mathbf{z})$  can therefore be written in the following form:

$$S(\mathbf{x}, \mathbf{y}, \mathbf{z}) = s_1(\mathbf{x}, \mathbf{z}) - s_2(\mathbf{y}, \mathbf{z}) \quad (58)$$

and applying the reverse transformation from  $\mathbf{x}, \mathbf{y}, \mathbf{z} \in D$  to  $P, Q, R \in \mathcal{P}$  implies the separability of the original scoring rule  $\mathcal{S}$ . ◀

### 2.2.3 Concluding the Proof

We can now conclude the section by proving the characterization theorem for retentive scoring rules. For the final proof, recall Definition 3 of symmetric retentive rules.

## 12:18 The Complexity of User Retention

**Proof of Theorem 4.** Given a proper scoring rule  $s : \mathcal{P} \times \mathcal{X} \rightarrow \mathbb{R}$  such that  $\mathcal{S}(P, Q, i) = s(P, i) - s(Q, i) - 1$ , we can apply Lemma 18 to show that  $\mathcal{S}(P, Q, i)$  is retentive. Conversely, given an analytic retentive scoring rule, we can apply Lemma 22 and obtain  $s_1, s_2$  such that  $\mathcal{S}(P, Q, i) = s_1(P, i) - s_2(Q, i)$ . The rule  $\mathcal{S}$  is retentive, and therefore satisfies (5). for all  $P \in \mathcal{P}$  and  $Q = P$  we obtain:

$$\mathcal{S}(P, P, i) = s_1(P, i) - s_2(P, i) = -1 \quad (59)$$

and therefore  $s_1(P, i) = s_2(P, i) - 1$  for all  $P$ , proving that  $\mathcal{S}$  is symmetric. ◀

### 3 Monotonicity

In this section we show that expected retention score in each round grows with the size of memory span, proving Theorem 6:

► **Theorem 6 (Knowledgeable Gurus Retain Better).** *Let  $\mathcal{S} : \mathcal{P} \times \mathcal{P} \times \mathcal{X} \rightarrow \mathbb{R}$  be an analytic retentive scoring rule, let  $G_1, G_2$  be two gurus with memory spans  $m_g^{(1)} \geq m_g^{(2)}$ . Then for any distribution  $\mathcal{T}$ , any coordinate  $x$ , and follower with memory span  $m_f \leq m_g^{(2)}$ :*

$$\mathbb{E}_{\mathcal{T}} [\mathcal{S}(P_1, Q, x)] \geq \mathbb{E}_{\mathcal{T}} [\mathcal{S}(P_2, Q, x)] \quad (11)$$

where  $P_1, P_2 \in \Delta(\mathcal{X})$  are the distributional forecasts of gurus  $G_1$  and  $G_2$  respectively, and  $Q \in \Delta(\mathcal{X})$  is the belief of the follower.

The proof will require a definition and a lemma: We first define the notion of *Localized Expected Gain* (Definition 24), which is a set function that quantifies the expected score for different choices of prior data. We then show that this function is monotone by proving Lemma 25, and use the result to prove the theorem itself.

#### Preliminaries

We denote the jointly distributed vector by  $(X_1, \dots, X_n) \sim \mathcal{T}$ . The marginal distribution of coordinate  $i$  is denoted by  $X_i$ . For  $t \in [n]$  and  $I \subseteq [n]$  such as  $t \notin I$ , the marginal value of coordinate  $t$  conditioned on the event  $X_{\setminus I} = x_{\setminus I}$  is denoted by  $(X_t | x_I)$ . When probability calculations are involved, we will omit the harpoon notation for brevity, and  $x_I$  and  $x_{\setminus I}$  will be used interchangeably.

► **Definition 24 (Localized Expected Gain).** Let  $(X_1, \dots, X_t) \sim D \in \Delta(\mathcal{X}^t)$  be a set of  $t$  jointly-distributed random variables, let  $I \subseteq [t-1]$ , and let  $s : \Delta(\mathcal{X}) \times \mathcal{X} \rightarrow \mathbb{R}$  be a proper (one-party) scoring rule. The *localized expected gain* is a set function  $f : 2^{[t-1]} \rightarrow \mathbb{R}$  defined as follows:

$$\forall I \subseteq [t-1] : f(I) \equiv \mathbb{E}_{(x_1, \dots, x_t) \sim D} [s((X_t | x_I), x_t)] \quad (60)$$

Intuitively, the localized expected gain function  $f(I)$  describes the expected score when  $X_{\setminus I}$  is being used as a prior. For example, for the log scoring rule  $s(P, i) = \log p_i$  defined in (3), the associated expected localized gain function is:

$$f_{\log}(I) = \sum_{x_I} \Pr(x_I) \sum_{x_t} \Pr(x_t | x_I) \log \Pr(x_t | x_I) = -H(X_t | X_I) \quad (61)$$

which is the additive inverse of the conditional entropy of  $X_t$  given  $X_I$ .

We now show that this function is also monotone for general proper scoring rules, which means that expected scores don't decrease when adding prior information, or "more knowledge doesn't hurt" regardless of the proper scoring rule being used:



► **Lemma 25.**  $f$  is a monotone set function:

$$\forall I \subseteq J \subseteq [t-1] : f(I) \leq f(J) \quad (62)$$

**Proof.** We start with the definition of  $f(I)$ :

$$f(I) = \mathbb{E}_{(x_1, \dots, x_t) \sim D} [s((X_t | x_I), x_t)] \quad (63)$$

$$= \sum_{x_{[t-1]}, x_t} \Pr(x_{[t-1]}, x_t) s((X_t | x_I), x_t) \quad (64)$$

[Decompose  $\Pr(x_{[t-1]}, x_t)$  using the law of total probability]

$$= \sum_{x_{[t-1]}, x_t} \Pr(x_J) \Pr(x_t | x_J) \Pr(x_{[t-1] \setminus J} | x_t, x_J) s((X_t | x_I), x_t) \quad (65)$$

[ $s$  does not depend on  $y_{[t] \setminus J}$ . Rearrange the summation]

$$= \sum_{x_J} \Pr(x_J) \sum_{x_t} \Pr(x_t | x_J) s((X_t | x_I), x_t) \sum_{x_{[t-1] \setminus J}} \Pr(x_{[t-1] \setminus J} | x_t, x_J) \quad (66)$$

[The rightmost factor is equal to 1]

$$= \sum_{x_J} \Pr(x_J) \sum_{x_t} \Pr(x_t | x_J) s((X_t | x_I), x_t) \quad (67)$$

Using the definition of expected one-party score defined in (19), we obtain that the rightmost factor in (67) is the expected score of  $P = (X_t | x_I)$  when the reference distribution is  $R = (X_t | x_J)$ :

$$f(I) = \sum_{x_J} \Pr(x_J) s((X_t | x_I), (X_t | x_J)) \quad (68)$$

We can now use the properness of  $s$  (see Definition 1) to obtain:

$$f(I) \leq \sum_{x_J} \Pr(x_J) s((X_t | x_J), (X_t | x_J)) \quad (69)$$

and apply steps (63),..., (67) in reverse order to obtain

$$\sum_{x_J} \Pr(x_J) s((X_t | x_J), (X_t | x_J)) = f(J) \quad (70)$$

proving that  $f(I) \leq f(J)$ . ◀

Using Lemma 25 we can generalize the result to retentive scoring rules, and prove the monotonicity theorem for retentive scoring rules:

**Proof of Theorem 6.** Guru 1 has memory span  $m_g^1$ , and therefore  $P_1 = (\mathcal{T} | u_{\downarrow I_1})$  such that  $|I_1| = m_g^1$ . Similarly, for guru 2 we obtain  $P_2 = (\mathcal{T} | u_{\downarrow I_2})$  such that  $|I_2| = m_g^2$  and for the follower  $Q = (\mathcal{T} | u_{\downarrow J})$  such that  $|J| = m_f$ .

$\mathcal{S}$  is analytic, and therefore symmetric according to Theorem 4. Denote  $\mathcal{S}(P, Q, i) = s(P, i) - s(Q, i) - 1$ . Taking the expectation over  $\mathcal{T}$  we obtain:

$$\mathbb{E}_{\mathcal{T}} [\mathcal{S}(P, Q, i)] = \mathbb{E}_{\mathcal{T}} [s(P, i)] - \mathbb{E}_{\mathcal{T}} [s(Q, i)] - 1 \quad (71)$$

Using Definition 24 we obtain:

$$\mathbb{E}_{\mathcal{T}} [\mathcal{S}(P, Q, i)] = f(I) - f(J) - 1 \quad (72)$$

## 12:20 The Complexity of User Retention

When  $m_g^1 \geq m_g^2$  and under the optimal choice of  $I_1$ , there exists  $I'_1$  such that  $I_2 \subseteq I'_1$  and  $f(I'_1) \leq f(I_1)$ . Applying Lemma 25 we obtain:

$$\mathbb{E}_{\mathcal{T}}[\mathcal{S}(P_1, Q, i)] = f(I_1) - f(J) - 1 \quad (73)$$

$$\geq f(I'_1) - f(J) - 1 \quad (74)$$

$$\geq f(I_2) - f(J) - 1 \quad (75)$$

$$= \mathbb{E}_{\mathcal{T}}[\mathcal{S}(P_2, Q, i)] \quad (76)$$

and therefore  $\mathbb{E}_{\mathcal{T}}[\mathcal{S}(P_1, Q, i)] \geq \mathbb{E}_{\mathcal{T}}[\mathcal{S}(P_2, Q, i)]$ .  $\blacktriangleleft$

### 4 The Binary Attributes Model

Under the Binary Attributes model, the universe of users is modeled using a  $k$ -dimensional linear subspace of  $\mathbb{F}_2^n$ .

$$U = \text{span}\{\bar{u}_1, \dots, \bar{u}_k\} \quad (77)$$

where  $\bar{u}_1, \dots, \bar{u}_k \in \mathbb{F}_2^n$  are a choice of basis vectors for the subspace. Under this realization of the Collaborative Discovery model, each user is represented using an  $n$ -dimensional binary vector, formally  $\mathcal{X}^n = \mathbb{F}_2^n$ .

#### Preliminaries

This section will assume familiarity with basic linear algebra over finite fields. A *view*  $I \subseteq [n]$  of a vector  $u \in \mathbb{F}_2^n$ , denoted by  $u_{\downarrow I}$ , is a linear projection of  $u$  to the subspace  $V_I = \text{span}\{e_i \mid i \in I\}$ . Similar to the previous section, we omit the harpoon notation when complex conditional probability expressions are involved. Given a vector space  $U$ , its *dual space* is defined as the set of linear constraints:  $U^\perp \equiv \{v \in \mathbb{F}_2^n \mid \forall u \in U : \langle u, v \rangle = 0\}$ . The *support* of a vector  $u \in U$  is the set of coordinates that contain non-zero elements:  $\text{support}(u) = \{i \mid u_i \neq 0\}$ . We denote the hamming distance of a vector  $u \in U$  by  $d(u) = |\text{support}(u)|$ . The hamming distance of the space  $U$  is defined as  $d(U) = \min_{u \in U \setminus \{0\}} d(u)$ .

#### 4.1 User Types as a Linear Subspace

We follow with a rigorous definition of the process under the Binary Attributes realization:

##### Initialization

At the start of the Collaborative Discovery process, the type of user  $u$  is picked uniformly from  $U$ , all the coordinates are undisclosed, and the initial retention parameter is  $r_0$ . We will denote the uniform random variable over the linear space by  $\mathcal{U} \sim \text{Uniform}(U)$ .

##### Prediction Rounds

During each round, the expert picks a coordinate  $i$  and provides a prediction distribution  $P \in \Delta(\{0, 1\})$  for its value. The retentive scoring function for this realization of the model is:

$$\mathcal{S}_{\text{bin}}(P, Q, x) = 2 \log_2 p_x - 2 \log_2 q_x - 1 \quad (78)$$

where  $x \in \{0, 1\}$ .  $\mathcal{S}_{\text{bin}}$  can be represented as  $\mathcal{S}_{\text{bin}}(P, Q, x) = s(Q, x) - s(P, x) - 1$ , where  $s(P, x) = 2 \log_2 p_x$  is the logarithmic scoring rule defined in (3), and therefore  $\mathcal{S}_{\text{bin}}$  is symmetric according to Definition 3.

We'll proceed to show that  $\mathcal{S}_{\text{bin}}$  has very intuitive properties. To do that, we start with a few basic claims about the structure of this probability space. The claims can be proved using basic linear algebra and probability. Proofs are included in Appendix B:

► **Claim 26.** *Let  $I \subseteq [n]$ . For every vector  $u_I \in U_I$ :*

$$\Pr(\mathcal{U}_I = u_I) = 2^{-\dim(U_I)} \quad (79)$$

For the following claim, recall that a *singleton distribution* is a probability distribution in which a single outcome has probability 1.

► **Claim 27.** *Let  $I \subseteq [n]$  and  $m \in [n] \setminus I$ , and assume a vector  $u \in \mathbb{F}_2^n$  has been picked uniformly at random from a vector space  $U$ .  $\Pr(u_m \mid u_I)$  is a singleton distribution if and only if  $e_m \in U^\perp_{|[n] \setminus I}$ .*

► **Claim 28.** *Let  $U$  be a linear space over  $\mathbb{F}_2^n$ , and let  $I \subseteq [n]$ ,  $m \in [n] \setminus I$ .  $e_m \in U^\perp_{|[n] \setminus I}$  if and only if  $\dim(U_{|I}) = \dim(U_{|I \cup \{m\}})$ .*

Using this framework, we now have enough tools to characterize the dynamics of scoring rule we defined:

► **Lemma 29** (Binary Attributes Scoring Rule Dynamics). *For a uniform distribution  $\mathcal{U}$  over a linear space  $U$  without constant bits, the retention score for a collaborative discovery process with infinite expert locality and zero layperson locality is given by:*

$$\mathcal{S}_{\text{bin}}((X_m \mid x_I), X_m, \mathcal{U}) = \begin{cases} 1 & e_m \in U^\perp_{|[n] \setminus I} \\ -1 & \text{otherwise} \end{cases} \quad (80)$$

$$= \begin{cases} 1 & \dim(U_{|I \cup \{m\}}) = \dim(U_{|I}) \\ -1 & \dim(U_{|I \cup \{m\}}) = \dim(U_{|I}) + 1 \end{cases} \quad (81)$$

**Proof.** When  $e_m \notin U^\perp_{|[n] \setminus I}$ , we get that  $\dim(U_{|I \cup \{m\}}) = 1$ , allowing us to apply Claim 26 and obtain  $\Pr(u_m = 0 \mid u_I) = \frac{1}{2}$ .

When  $e_m \in U^\perp_{|[n] \setminus I}$  there exists  $v \in U^\perp$ ,  $I' \subseteq I$  such that  $\text{support}(v) = I' \cup \{m\}$ . Claim 27 implies that  $u_m$  is determined given  $u_I$ .

Combining the results, we obtain for all  $I \subseteq [n]$ ,  $m \notin I$ :

$$\Pr(u_m = 0 \mid u_I) \in \begin{cases} \{0, 1\} & e_m \in U^\perp_{|[n] \setminus I} \\ \{\frac{1}{2}\} & \text{otherwise} \end{cases} \quad (82)$$

There are no constant bits in  $U$ , and therefore  $\dim U_{|I \cup \{m\}} = 1$  for all  $m \in [n]$ . By Claim 26 we obtain that the marginal distribution for each coordinate is uniform, and therefore a layperson with zero locality will always predict a uniform distribution.

Plugging (82) into the definition of  $\mathcal{S}_{\text{bin}}$  in equation (78), the score for the first case is  $\log_2 \frac{1}{2 \cdot 0.5^2} = 1$ , and the score for the second case is  $\log_2 \frac{0.5^2}{2 \cdot 0.5^2} = -1$ , leading to equation (80). The transition from (80) to (81) is given by Claim 28. ◀

## 4.2 Retention Complexity of Linear Codes

We will now apply the notion of retention complexity introduced in Definition 8 to the Binary Attributes model. We will first show that there exist non-trivial upper and lower bounds for retention complexity in this realization of the Collaborative Discovery model, and then show that the bounds are tight. Recall Lemma 11:

► **Lemma 30** (Retention Complexity Bounds for Linear Spaces). *For a uniform distribution  $\mathcal{U}$  over a linear space  $U \subseteq \mathbb{F}_n^2$  with unbounded guru memory span and zero follower memory span, the retention complexity satisfies:*

$$d(U^\perp) - 1 \leq r_{(\mathcal{S}_{\text{bin}}, \infty, 0)}(\mathcal{U}) \leq \dim(U) \quad (14)$$

**Proof of Lemma 11.** The retention parameter at the end of each round  $t$  is defined according to equation (1):

$$r_t = r_0 + \sum_{i=1}^t \mathcal{S}_{\text{bin}}((X_{\sigma_i} | x_{I_i}), X_{\sigma_i}, \mathcal{U}) \quad (83)$$

For the lower bound, observe that  $U^\perp_{|[n] \setminus I_t}$  does not contain any singleton element when  $|I_t| \leq d(U^\perp) - 2$ . Since  $|I_t| \leq t - 1$  by definition, we can combine the inequalities and obtain that no punctured-dual-space singleton exists when  $t \leq d(U^\perp) - 1$ . We can now apply Lemma 29 and obtain that  $\mathcal{S}_{\text{bin}}((X_{\sigma_i} | x_{I_i}), X_{\sigma_i}, \mathcal{U}) = -1$  for all  $i \in \{1, \dots, t\}$ . Plugging into the retention parameter at time  $t = d(U^\perp) - 1$ :

$$r_t = r_0 + \sum_{i=1}^{d(U^\perp)-1} (-1) = r_0 - (d(U^\perp) - 1) \quad (84)$$

And the positivity constraint on  $r_t$  implies that  $r_0 \geq (d(U^\perp) - 1)$ .

For the upper bound, assume without loss of generality that the first  $k = \dim(U)$  coordinates of  $U$  are linearly independent, and set  $\sigma_i = i, I_i = \{1, \dots, (i - 1)\}$  for all  $i \in \{1, \dots, k\}$ . Observe that:

$$\dim(U_{|I_i}) = \begin{cases} i - 1 & 1 \leq i \leq k \\ k & k < i \end{cases} \quad (85)$$

Applying Lemma 29 we get:

$$\mathcal{S}_{\text{bin}}((X_{\sigma_i} | x_{I_i}), X_{\sigma_i}, \mathcal{U}) = \begin{cases} -1 & 1 \leq i \leq k \\ 1 & k < i \end{cases} \quad (86)$$

Hence for  $r_0 = k$  we get  $r_t \geq 0$  for all  $t \in \{1, \dots, n\}$ . ◀

In the asymptotic setting it is common to consider  $n, k \rightarrow \infty$ . In this case,  $d(U^\perp)$  can stay constant, forming a large gap between the bounds. We will proceed to show that the upper and lower bounds are indeed tight in the asymptotic setting.

### 4.2.1 Linear Functions are Retentively Learnable

Let  $n = 2^k - 1$ . Given a binary message  $x \in \{0, 1\}^k$ , the *Walsh-Hadamard* code (WH) encodes the message into a codeword  $\text{WH}(x)$  using an encoding function  $\text{WH} : \{0, 1\}^k \rightarrow \{0, 1\}^n$ , such that for every  $y \in (\{0, 1\}^k \setminus \{0^k\})$ , the  $y$ th coordinate of  $\text{WH}(x)$  is equal to  $(x \cdot y)$ .

Walsh-Hadamard is the space of linear functions over  $\mathbb{F}_2^k$ . Note that we slightly deviate from the common definition by omitting the 0th coordinate which is always equal to zero. It is a  $[2^k - 1, k, 2^{k-1}]_2$  locally-correctable code with  $q = 2$  queries. See [1] for a thorough discussion of Walsh-Hadamard codes and its applications in theoretical computer science.

We will show that a uniform distribution over the  $k$ -dimensional WH code achieves the retention complexity lower bound for all  $k \in \mathbb{N}$ . Recall Lemma 12:

► **Lemma 31** (Walsh-Hadamard Retention Complexity). *For all  $k \in \mathbb{N}$ , a  $k$ -dimensional Walsh-Hadamard code satisfies:*

$$r_{(\mathcal{S}_{\text{bin}}, 2, 0)}(\text{WH}) = 2 \quad (15)$$

In order to prove the lemma, we first characterize the constraints of the WH code (Claim 32, Claim 33), and then use the results to construct an explicit formula for the retention score when the  $\mathcal{S}_{\text{bin}}$  retentive score rule is being used (Lemma 34), giving an upper bound for  $r_{(\mathcal{S}_{\text{bin}}, 3, 0)}(\text{WH})$  which is equal to the lower bound we established in Lemma 11. Proofs for the claims can be found in Appendix B.

► **Claim 32.** *Let  $y^{(1)}, \dots, y^{(m)} \in (\{0, 1\}^k \setminus \{0^k\})$ .*

$$\left( \sum_{i=1}^m e_{y^{(i)}} \right) \in \text{WH}^\perp \iff \sum_{i=1}^m y^{(i)} = 0 \quad (87)$$

► **Claim 33.**

$$d(\text{WH}^\perp) = 3 \quad (88)$$

► **Lemma 34.** *Let  $k > 0$ , and let  $u$  be a uniformly-sampled vector from the  $k$ -dimensional WH code. Set a natural ordering over the coordinates ( $\gamma_t = t$  for  $t \in \{1, \dots, 2^k - 1\}$ ), and set a sequence of subsets:*

$$I_t = \begin{cases} \{2^{\lfloor \log_2 t \rfloor}, t - 2^{\lfloor \log_2 t \rfloor}\} & t > 2, \text{ and } t \text{ is not a power of } 2 \\ \emptyset & \text{otherwise} \end{cases}$$

*For collaborative discovery with respect to  $\mathcal{S} = \mathcal{S}_{\text{bin}}$ ,  $m_g = 2$ ,  $m_f = 0$  and  $r_0 = 2$ , the ordering  $\gamma_t$  and sequence of sets  $I_t$  satisfies:*

$$r_t = t - 2^{\lfloor \log_2 t \rfloor} \quad (89)$$

*for all  $1 \leq t < 2^k$ .*

**Proof.** By induction:

For the base case  $t \in \{1, 2\}$ . According to the definition,  $\gamma_1 = 1, \gamma_2 = 2$ , and  $I_1 = I_2 = \emptyset$ . We use Claim 33 and an argument similar to the one in Lemma 11 to show that there's no singleton in the punctured dual-space in the first two rounds. The guess in the first two rounds will therefore be a uniform one, and  $r_1 = 1, r_2 = 0$ . Indeed we can substitute 0, 1 into (89) see that  $1 - 2^{\lfloor \log_2 1 \rfloor} = 1$  and  $1 - 2^{\lfloor \log_2 2 \rfloor} = 1$ .

For  $t > 2$ , assume the retention parameter formula holds for  $t - 1$ , and consider the two following cases:

## 12:24 The Complexity of User Retention

- When  $t$  is not a power of two, it can be represented as the XOR between two preceding coordinates, for example  $t' = 2^{\lfloor \log_2 t \rfloor}$  and  $t'' = t - 2^{\lfloor \log_2 t \rfloor}$ . Note that  $I_t = \{t', t''\}$ , and that  $|I_t| = 2$ , satisfying the  $m_g = 2$  memory span constraint. Using Claim 32 we obtain that  $e_t$  is a singleton in the punctured dual-space, and therefore  $r_t = r_{t-1} + 1$  by Lemma 29. Using the induction hypothesis and the fact that  $\lfloor \log_2 t \rfloor = \lfloor \log_2 (t-1) \rfloor$  when  $t$  is not a power of two, we obtain:

$$\begin{aligned} r_t &= r_{t-1} + 1 \\ &= (t-1) - 2\lfloor \log_2 (t-1) \rfloor + 1 \\ &= t - 2\lfloor \log_2 t \rfloor \end{aligned}$$

- When  $t$  is a power of two, it cannot be represented as the XOR between preceding coordinates, as for all of them the index of the most significant bit is strictly less than  $\log_2 t$ . By Lemma 29 we obtain that  $r_t = r_{t-1} - 1$ , and using the fact that  $\lfloor \log_2 t \rfloor = \lfloor \log_2 (t-1) \rfloor + 1$  when  $t$  is a power of two we indeed get:

$$\begin{aligned} r_t &= r_{t-1} - 1 \\ &= (t-1) - 2\lfloor \log_2 (t-1) \rfloor - 1 \\ &= t - 2\lfloor \log_2 t \rfloor \end{aligned} \quad \blacktriangleleft$$

► **Remark (Non-punctured Walsh-Hadamard).** In the non-punctured Walsh-Hadamard code, the 0th coordinate is not omitted, and always equal to zero. Offsetting the sequences in Lemma 34 can show that the same upper bound also holds for the non-punctured version of the Walsh-Hadamard code.

Combining the results proves Lemma 12:

**Proof of Lemma 12.** Lemma 34 shows an upper bound of 2 for the retention complexity of WH. Lemma 11, together Claim 33, tells us that this is also the lower bound for the retention complexity in this case, and therefore  $r_{(\mathcal{S}_{\text{bin}}, 2, 0)}(\text{WH}) = 2$ .  $\blacktriangleleft$

We can now conclude and prove Theorem 10. Recall the theorem statement:

► **Theorem 10 (Linear functions are retentively learnable).** *The property of linear functions over the two-element field  $\mathbb{F}_2$  is retentively learnable.*

**Proof of Theorem 10.** For linear functions over  $\mathbb{F}_2^n$ , the corresponding family of categorical distributions is  $\mathbb{D} = \{\mathcal{U}_n\}_{n=2^k}$ , where  $\mathcal{U}_i$  is the family of uniform distributions over the non-punctured Walsh-Hadamard code. Lemma 34 shows that each  $\mathcal{U}_i$  is retainable with respect to  $(\mathcal{S}_{\text{bin}}, 2, 0, 2)$ .  $\blacktriangleleft$

### 4.2.2 Random LDPC Codes are Asymptotically Hard to Retain

Let  $G = (L, R, E)$  be a bipartite multigraph with  $|L| = n$ ,  $|R| = m$ . Associate a distinct Boolean variable  $x_i$  with any  $i \in L$ . For each  $j \in R$ , let  $N(j) \subseteq L$  be the set of neighbors of  $j$ . The  $j$ th constraint is  $A_j(x_1, \dots, x_n) = \sum_{i \in N(j)} x_i \pmod 2$ . The code defined by  $G$  is:

$$\mathcal{C}(G) = \{x \in \{0, 1\}^n \mid \forall j \in [m] : A_j(x) = 0\}$$

A random  $(c, d)$ -regular LDPC code of length  $n$  is obtained by taking  $\mathcal{C}(G)$  for a random  $(c, d)$ -regular  $G$  with  $n$  left vertices. Random LDPC codes were first described and analyzed by [10]. We will show that a randomly chosen LDPC code asymptotically achieves the upper bound for retention complexity with high probability. Recall Theorem 13:

► **Theorem 13** (LDPC Retention Complexity). *For a proper choice of constants  $c, d > 0$  and sufficiently large  $n$ , the retention complexity of a random  $(c, d)$ -regular LDPC code over  $\mathbb{F}_2^n$  is linear with high probability:*

$$r_{(\mathcal{S}_{\text{bin}, \infty, 0})}(\text{LDPC}) \stackrel{\text{w.h.p.}}{=} \Omega(\dim(\text{LDPC})) \quad (16)$$

► **Definition 35** ( $(q, \mu)$  code locality, [4]). A linear space  $V$  is  $(q, \mu)$ -local if every  $v \in V$  that is a sum of at least  $\mu m$  basis vectors has  $d(v) \geq q$ .

The following lemma shows that a random LDPC code has  $(q, \mu)$ -locality with high probability for a proper choice of parameters:

► **Lemma 36** ([4], Lemma 3.6). *Fix odd integer  $c \geq 7$  and constants  $\mu, \delta, d > 0$  satisfying:*

$$\mu \leq \frac{c^{-2}}{100}; \quad \delta < \mu^c; \quad d > \frac{2\mu c^2}{(\mu^c - \delta)^2} \quad (90)$$

*Then, for all sufficiently large  $n$ , with high probability for a random  $(c, d)$ -regular graph  $G$  with  $n$  left vertices and  $m = \frac{c}{d}n$  right vertices, the corresponding LDPC code  $\mathcal{C}(G)$  is linearly-independent, and  $(\delta n, \mu)$ -local.*

► **Remark 37** (A Proper Choice of Parameters). For our proof of Theorem 13, the constants in (90) need be chosen such that  $\delta - \frac{2\mu c}{d} \geq 0$ .

Such a choice of random code parameters is indeed possible: For example, by fixing  $c \geq 7$  and taking  $\mu = \frac{c^{-2}}{100}$ ,  $\delta = (\mu^c - \varepsilon_0)$ ,  $d = \frac{8\mu c^2}{(\mu^c - \delta)^2}$  we get:

$$\delta - 2\frac{\mu c}{d} = \mu^c - \varepsilon_0 - 2\frac{\mu c}{\frac{8\mu c^2}{(\mu^c - \delta)^2}} = \mu^c - \varepsilon_0 - \frac{\varepsilon_0^2}{4c}$$

Which is strictly larger than zero for all  $0 < \varepsilon_0 < 2c\left(\sqrt{1 + \frac{\mu^c}{c}} - 1\right)$ .

We now use this to prove Theorem 13:

**Proof of Theorem 13.** Fix odd integer  $c \geq 7$  and constants  $\mu, \varepsilon, \delta, d > 0$  satisfying equation (90) and  $\delta \geq \frac{\mu c}{d}$ . See Remark 37 for a specific choice of such constants. Let  $V$  be a random LDPC code of dimension  $n$  corresponding to this choice of constants. Assume that  $n$  is large enough to satisfy Lemma 36. Assume by contradiction that  $r_0 \leq n\left(\delta - \frac{2\mu c}{d}\right) - 1$ , and the Collaborative Discovery process lasts until round  $n$ . Set  $t = \lfloor \delta n \rfloor - 1$ .

At the end of round  $t$ , the coordinates  $I_t \subseteq [n]$  are disclosed. Denote by  $n_-$  the number of times a uniform distribution was predicted by the expert. Using Lemma 29, the total retention accumulated at the end of round  $t$  is equal to:

$$r_t = r_0 - n_- + (t - n_-) \quad (91)$$

$r_t \geq 0$ , and therefore  $n_- \leq \frac{t+r_0}{2}$ . Using Lemma 29 again, we obtain that  $n_- = \dim(V_{\perp I_t})$ . In addition, the dimensions of a vector space and its dual sum up to  $t$ , hence  $\dim\left((V_{\perp I_t})^\perp\right) = (t - n_-) \geq \frac{t-r_0}{2}$ . This gives us a lower bound for  $\dim\left((V_{\perp I_t})^\perp\right)$ .

$(V_{\perp I_t})^\perp$  consists of vectors  $v \in V^\perp$  such that  $\text{support}(v) \subseteq I_t$ . The conditions of Lemma 36 are satisfied by our choice of constants, and we can apply it to obtain that  $V^\perp$  of the random code we picked is  $(\delta n, \mu)$ -local with high probability, and therefore every  $v \in V^\perp$  that is a

sum of at least  $\frac{c}{d}\mu n$  dual basis vectors has  $d(v) \geq \delta n$ . For  $t < \delta n$ , all the vectors of  $(V_{I_t})^\perp$  are a sum of  $\frac{c}{d}\mu n$  basis vectors at most, hence  $\dim(V_{I_t}^\perp) \leq \frac{c}{d}\mu n$ , implying an upper bound for  $\dim((V_{I_t})^\perp)$ .

Combining the bounds we obtain:

$$\frac{t - r_0}{2} \leq \dim((V_{I_t})^\perp) < \frac{c}{d}\mu n \quad (92)$$

For  $t = \lfloor \delta n \rfloor - 1$  and  $r_0 \leq n(\delta - \frac{2\mu c}{d}) - 1$  we have:

$$\frac{t - r_0}{2} \geq \frac{(\delta n - 1) - (n(\delta - \frac{2\mu c}{d}) - 1)}{2} = \frac{c}{d}\mu n \quad (93)$$

Leading to a contradiction, since the lower bound in equation (92) must be greater than the upper bound. From this we get  $r_0 > n(\delta - \frac{2\mu c}{d})$ , and therefore  $r_0 = \Omega(n) = \Omega(\dim(V))$ . ◀

---

## References

- 1 Sanjeev Arora and Boaz Barak. *Computational complexity: a modern approach*. Cambridge University Press, 2009.
- 2 Amir Ban and Nati Linial. The dynamics of reputation systems. In *Proceedings of the 13th Conference on Theoretical Aspects of Rationality and Knowledge*, pages 91–100. ACM, 2011.
- 3 Ayelet Ben-Sasson, Eli Ben-Sasson, Kayla Jacobs, and Eden Saig. Baby CROINC: An Online, Crowd-based, Expert-curated System for Monitoring Child Development. In *Proceedings of the 11th EAI International Conference on Pervasive Computing Technologies for Healthcare, PervasiveHealth '17*, pages 110–119, New York, NY, USA, 2017. ACM. doi:10.1145/3154862.3154887.
- 4 Eli Ben-Sasson, Prahladh Harsha, and Sofya Raskhodnikova. Some 3CNF properties are hard to test. *SIAM Journal on Computing*, 35(1):1–21, 2005.
- 5 Manuel Blum, Michael Luby, and Ronitt Rubinfeld. Self-Testing/Correcting with Applications to Numerical Problems. In *Proceedings of the 22nd Annual ACM Symposium on Theory of Computing, May 13-17, 1990, Baltimore, Maryland, USA*, pages 73–83, 1990. doi:10.1145/100216.100225.
- 6 Glenn W Brier. Verification of forecasts expressed in terms of probability. *Monthly weather review*, 78(1):1–3, 1950.
- 7 Kam Tong Chan, Irwin King, and Man-Ching Yuen. Mathematical modeling of social games. In *Computational Science and Engineering, 2009. CSE'09. International Conference on*, volume 4, pages 1205–1210. IEEE, 2009.
- 8 Alexander Philip Dawid and Monica Musio. Theory and applications of proper scoring rules. *Metron*, 72(2):169–183, 2014.
- 9 Gideon Dror, Dan Pelleg, Oleg Rokhlenko, and Idan Szpektor. Churn prediction in new users of Yahoo! answers. In *Proceedings of the 21st International Conference on World Wide Web*, pages 829–834. ACM, 2012.
- 10 Robert Gallager. Low-density parity-check codes. *IRE Transactions on information theory*, 8(1):21–28, 1962.
- 11 Tilmann Gneiting and Adrian E Raftery. Strictly proper scoring rules, prediction, and estimation. *Journal of the American Statistical Association*, 102(477):359–378, 2007.
- 12 Michael H. Goldhaber. The attention economy and the Net. *First Monday*, 2(4), 1997. doi:10.5210/fm.v2i4.519.
- 13 Oded Goldreich, Shafi Goldwasser, and Dana Ron. Property testing and its connection to learning and approximation. *Journal of the ACM (JACM)*, 45(4):653–750, 1998.



- 14 Gradient Theorem. Gradient Theorem — Wikipedia, The Free Encyclopedia, 2017. [Online; accessed 08-September-2017]. URL: [https://en.wikipedia.org/w/index.php?title=Gradient\\_theorem&oldid=781791224](https://en.wikipedia.org/w/index.php?title=Gradient_theorem&oldid=781791224).
- 15 Arlo D Hendrickson and Robert J Buehler. Proper scores for probability forecasters. *The Annals of Mathematical Statistics*, pages 1916–1921, 1971.
- 16 Richard A Lanham. *The economics of attention: Style and substance in the age of information*. University of Chicago Press, 2006.
- 17 John McCarthy. Measures of the value of information. *Proceedings of the National Academy of Sciences*, 42(9):654–655, 1956.
- 18 George A Miller. The magical number seven, plus or minus two: some limits on our capacity for processing information. *Psychological review*, 63(2):81, 1956.
- 19 Yehuda Pinchover and Jacob Rubinstein. *An introduction to partial differential equations*. Cambridge university press, 2005.
- 20 Paul Resnick, Ko Kuwabara, Richard Zeckhauser, and Eric Friedman. Reputation systems. *Communications of the ACM*, 43(12):45–48, 2000.
- 21 Paul Resnick and Richard Zeckhauser. Trust among strangers in Internet transactions: Empirical analysis of eBay’s reputation system. In *The Economics of the Internet and E-commerce*, pages 127–157. Emerald Group Publishing Limited, 2002.
- 22 Ralph Tyrell Rockafellar. *Convex analysis*. Princeton university press, 2015.
- 23 Leonard J Savage. Elicitation of personal probabilities and expectations. *Journal of the American Statistical Association*, 66(336):783–801, 1971.
- 24 Endel Tulving and Fergus IM Craik. *The Oxford handbook of memory*. Oxford: Oxford University Press, 2000.
- 25 Leslie G Valiant. A theory of the learnable. *Communications of the ACM*, 27(11):1134–1142, 1984.
- 26 Chih-Ping Wei and I-Tang Chiu. Turning telecommunications call details to churn prediction: a data mining approach. *Expert systems with applications*, 23(2):103–112, 2002.

## A Retentive Scoring Appendices

► **Claim 21.** Let  $D \subseteq \mathbb{R}^n$  such that  $\mathbf{x}, \mathbf{y} \in D$ . For every analytic function  $u : D \times D \rightarrow \mathbb{R}$  satisfying the equation

$$u(\mathbf{x}, \mathbf{y}) - \sum_{i=1}^n (y_i - x_i) \frac{\partial u(\mathbf{x}, \mathbf{y})}{\partial x_i} = 0 \quad (32)$$

there exist functions  $\alpha_1, \dots, \alpha_n : D \rightarrow \mathbb{R}$  such that:

$$u(\mathbf{x}, \mathbf{y}) = \sum_{i=1}^n \alpha_i(\mathbf{y})(y_i - x_i) \quad (33)$$

**Proof of Claim 21.**  $u(\mathbf{x}, \mathbf{y})$  is analytic in  $D$ , and therefore it has a unique representation as a convergent power series about  $(\mathbf{y}, \mathbf{y})$ :

$$u(\mathbf{x}) = \sum_{j_1, \dots, j_{2n}=0}^{\infty} c_{j_1, \dots, j_{2n}} \prod_{k=1}^n (y_k - x_k)^{j_k} \prod_{k'=n+1}^{2n} y_{k'}^{j_{k'}} \quad (94)$$

Note that  $(y - x) \frac{\partial (y-x)^a}{\partial x} = -a(y-x)^a$  for all  $a \in \mathbb{R}$ , and therefore:

$$\sum_{i=1}^n (y_i - x_i) \frac{\partial}{\partial x_i} \prod_{k=1}^n (y_k - x_k)^{j_k} = - \sum_{i=1}^n j_i \prod_{k=1}^n (y_k - x_k)^{j_k} \quad (95)$$

Using the above, we obtain for (32):

$$0 = u + \sum_{i=1}^n (y_i - x_i) \frac{\partial u}{\partial x_i} \quad (96)$$

[Use (94) to represent the rightmost term as a power series]

$$= u + \sum_{i=1}^n (y_i - x_i) \frac{\partial}{\partial x_i} \left( \sum_{j_1, \dots, j_{2n}=0}^{\infty} c_{j_1, \dots, j_{2n}} \prod_{k=1}^n (y_k - x_k)^{j_k} \prod_{k'=n+1}^{2n} y_{k'}^{j_{k'}} \right) \quad (97)$$

[Derivative operator does not affect the factors that don't depend on  $x$ ]

$$= u + \sum_{j_1, \dots, j_{2n}=0}^{\infty} c_{j_1, \dots, j_{2n}} \prod_{k'=n+1}^{2n} y_{k'}^{j_{k'}} \left( \sum_{i=1}^n (y_i - x_i) \frac{\partial}{\partial x_i} \prod_{k=1}^n (y_k - x_k)^{j_k} \right) \quad (98)$$

[Apply the derivative using (95)]

$$= u + \sum_{j_1, \dots, j_{2n}=0}^{\infty} c_{j_1, \dots, j_{2n}} \prod_{k'=n+1}^{2n} y_{k'}^{j_{k'}} \left( - \sum_{i=1}^n j_i \right) \prod_{k=1}^n (y_k - x_k)^{j_k} \quad (99)$$

[Use (94) to represent the leftmost term as a power series]

$$= \sum_{j_1, \dots, j_{2n}=0}^{\infty} c_{j_1, \dots, j_{2n}} \left( 1 - \sum_{i=1}^n j_i \right) \prod_{k'=n+1}^{2n} y_{k'}^{j_{k'}} \prod_{k=1}^n (y_k - x_k)^{j_k} \quad (100)$$

If a convergent power series is equal to zero, then all its coefficients must be equal to zero as well. From (100) we obtain:

$$\forall j_1, \dots, j_n \in \mathbb{N} : c_{j_1, \dots, j_n} \left( 1 - \sum_{i=1}^n j_i \right) = 0 \quad (101)$$

Therefore  $c_{j_1, \dots, j_n} = 0$  when  $\sum_{i=1}^n j_i \neq 1$ , and analytic solutions for (32) can only contain linear coefficients of  $(y_i - x_i)$  in their series expansion. Let  $k \in [n]$ . when  $j_k = 1$  we denote  $c_{j_1, \dots, j_{2n}} \equiv c_{k, j_{n+1}, \dots, j_{2n}}$ . Plug back into the series representation (94) to obtain:

$$u(\mathbf{x}) = \sum_{i=1}^n \left( \sum_{j_{n+1}, \dots, j_{2n}=0}^{\infty} c_{i, j_{n+1}, \dots, j_{2n}} \prod_{k'=n+1}^{2n} y_{k'}^{j_{k'}} \right) (y_i - x_i) \quad (102)$$

Denoting  $\alpha_i(\mathbf{y}) \equiv \left( \sum_{j_{n+1}, \dots, j_{2n}=0}^{\infty} c_{i, j_{n+1}, \dots, j_{2n}} \prod_{k'=n+1}^{2n} y_{k'}^{j_{k'}} \right)$  leads to the linear representation of  $u$  in (33). ◀

## B Binary Attributes Appendices

### B.1 The Binary Attributes Model

► **Claim 26.** Let  $I \subseteq [n]$ . For every vector  $u_I \in U_I$ :

$$\Pr(\mathcal{U}_I = u_I) = 2^{-\dim(U_I)} \quad (79)$$

**Proof of Claim 26.** Without loss of generality assume that  $I = \{1, \dots, |I|\}$ , and choose a basis  $U = \text{span}\{\bar{u}_1, \dots, \bar{u}_k\}$  which is diagonalized. Each vector in  $U$  can be represented as linear combination of basis elements. By definition, only only the first  $\dim(U_I)$  diagonalized

basis vectors have support in  $I$ , and therefore every vector in  $U_I$  can be written as a linear combination of the view of the first  $\dim U_I$  basis vectors of  $U$ :

$$\forall u_I \in U_I, \exists \alpha_1, \dots, \alpha_{\dim(U_I)} : u_I = \sum_{i=1}^{\dim(U_I)} \alpha_i (\bar{u}_i)_{\downarrow I} \quad (103)$$

Picking  $u$  at random is equivalent to choosing each  $\alpha_i$  uniformly, or equivalently, picking  $(\alpha_1, \dots, \alpha_{\dim(U_I)}) \sim \text{Uniform}(\{0, 1\}^{\dim(U_I)})$ . From this correspondence it follows that  $\Pr(u_I) = \Pr(\alpha_1, \dots, \alpha_{\dim(U_I)}) = 2^{-\dim(U_I)}$ . ◀

► **Claim 27.** Let  $I \subseteq [n]$  and  $m \in [n] \setminus I$ , and assume a vector  $u \in \mathbb{F}_2^n$  has been picked uniformly at random from a vector space  $U$ .  $\Pr(u_m \mid u_I)$  is a singleton distribution if and only if  $e_m \in U^\perp_{\downarrow [n] \setminus I}$ .

**Proof of Claim 27.** When  $e_m \in U^\perp_{\downarrow [n] \setminus I}$  there exists a vector  $v \in U^\perp$  and  $I' \subseteq I$  such that  $\text{support}(v) = \{m\} \cup I'$ .  $v$  is a dual-space vector, and therefore  $\sum_{i \in I'} u_i + u_m = 0$ . The value  $u_m \in \{0, 1\}$  is completely determined by the values of  $u_{I'}$ , and therefore  $\Pr(u_m \mid u_I)$  is a singleton distribution.

Conversely, observe that restricting a vector to a subset of coordinates  $I \subseteq [n]$  can be viewed as a linear projection operation  $P_I \equiv \sum_{i \in I} e_i e_i^T$ . Let  $v \in U$  be a vector for which  $v_I = u_I$ . The set of vectors  $u' \in U$  for which  $u'_I = u_I$  is an affine subspace  $U'$  of  $U$ :

$$U' = v + V' = \{v + v' \mid v' \in U, P_I v' = 0\} \quad (104)$$

Note that  $V'$  is a linear subspace of  $U$ , and therefore:

$$(V')^\perp = \text{span}(U^\perp \cup \{e_i \mid i \in I\}) \quad (105)$$

Using the assumption that  $\Pr(u_m \mid u_I)$  is a singleton distribution, we get that the  $m$ -th coordinate is constant in  $U'$ , and therefore  $P_{\{m\}} V' = 0$ , and  $e_m \in (V')^\perp$ . denote  $U^\perp = \text{span}\{\bar{u}_1^\perp, \dots, \bar{u}_{n-k}^\perp\}$ . Using (105) we can write  $e_m$  as a linear combination of spanning set elements:

$$e_m = \sum_{i=1}^{|I|} \alpha_i e_i + \sum_{j=1}^{n-k} \beta_j \bar{u}_j^\perp \quad (106)$$

Restricting the view to coordinates  $[n] \setminus I$ , the terms in the first sum vanish, yielding:

$$e_m = P_{[n] \setminus I} e_m = \sum_{j=1}^{n-k} \beta_j P_{[n] \setminus I} \bar{u}_j^\perp \quad (107)$$

We have shown that it's possible to write  $e_m$  as a linear combination of punctured dual space elements, hence  $e_m \in U^\perp_{\downarrow [n] \setminus I}$ . ◀

► **Claim 28.** Let  $U$  be a linear space over  $\mathbb{F}_n^2$ , and let  $I \subseteq [n]$ ,  $m \in [n] \setminus I$ .  $e_m \in U^\perp_{\downarrow [n] \setminus I}$  if and only if  $\dim(U_{\downarrow I}) = \dim(U_{\downarrow I \cup \{m\}})$ .

**Proof of Claim 28.** Assume a uniform distribution over  $U$ , then  $e_m \in U^\perp_{\downarrow [n] \setminus I}$ , if and only if  $\Pr(u_m \mid u_I)$  is a singleton distribution by Claim 27.

According to the law of total probability,  $\Pr(u_m \mid u_I)$  is a singleton distribution if and only if the following marginal distributions are equal:  $\Pr(u_{I \cup \{m\}}) = \Pr(u_I)$ .

Using Claim 26 we obtain that the two probabilities are equal if and only if  $\dim(U_{\downarrow I}) = \dim(U_{\downarrow I \cup \{m\}})$ . ◀

## B.2 Retention Complexity of the Walsh-Hadamard Code

► **Claim 32.** Let  $y^{(1)}, \dots, y^{(m)} \in (\{0, 1\}^k \setminus \{0^k\})$ .

$$\left( \sum_{i=1}^m e_{y^{(i)}} \right) \in \text{WH}^\perp \iff \sum_{i=1}^m y^{(i)} = 0 \quad (87)$$

**Proof of Claim 32.** By definition,  $(\sum_{i=1}^m e_{y^{(i)}}) \in \text{WH}^\perp$  if and only if  $(\sum_{i=1}^m e_{y^{(i)}}) \cdot u = 0$  for all  $u \in \text{WH}$ . For an arbitrary  $u$ , let  $w \in \{0, 1\}^k$  such that  $u = \text{WH}(w)$ . Plug into the definition of WH and obtain:

$$\begin{aligned} \left( \sum_{i=1}^m e_{y^{(i)}} \right) \cdot u &= \sum_{i=1}^m u_{y^{(i)}} \\ &= \sum_{i=1}^m w \cdot y^{(i)} \\ &= w \cdot \left( \sum_{i=1}^m y^{(i)} \right) \end{aligned}$$

Observe that the inner product is equal to zero for all  $u \in \text{WH}$  if and only if  $w \cdot (\sum_{i=1}^m y^{(i)}) = 0$  for all  $w \in \{0, 1\}^k$ . This happens if and only if  $(\sum_{i=1}^m y^{(i)}) = 0$ , proving our claim. ◀

► **Claim 33.**

$$d(\text{WH}^\perp) = 3 \quad (88)$$

**Proof of Claim 33.** By Claim 32, the vectors corresponding to the support of each constraint in  $\text{WH}^\perp$  must have their XORs equal to zero.

$0^k \notin (\{0, 1\}^k \setminus \{0^k\})$ , and therefore there are no constraints of size 1, and we have  $d(\text{WH}^\perp) > 1$ . Similarly, for all  $x, y \in (\{0, 1\}^k \setminus \{0^k\})$  such that  $x \neq y$  we get  $x + y \neq 0$ , and therefore there are no constraints of size 2, and  $d(\text{WH}^\perp) > 2$ .

Taking  $x \neq y$  and  $z = x + y$  gives 3 coordinates with corresponding vectors that sum up to zero, and therefore  $d(\text{WH}^\perp) \leq 3$  according to Claim 32. Combining the conclusions we obtain  $d(\text{WH}^\perp) = 3$ . ◀

# Torus Polynomials: An Algebraic Approach to ACC Lower Bounds

**Abhishek Bhrushundi**<sup>1</sup>

Rutgers University, New Brunswick, USA  
abhishek.bhr@rutgers.edu

**Kaave Hosseini**

University of California, San Diego, USA  
skhossei@ucsd.edu

**Shachar Lovett**

University of California, San Diego, USA  
slovett@ucsd.edu

**Sankeerth Rao**

University of California, San Diego, USA  
skaringu@ucsd.edu

---

## Abstract

We propose an algebraic approach to proving circuit lower bounds for  $\text{ACC}^0$  by defining and studying the notion of *torus polynomials*. We show how currently known polynomial-based approximation results for  $\text{AC}^0$  and  $\text{ACC}^0$  can be reformulated in this framework, implying that  $\text{ACC}^0$  can be approximated by low-degree torus polynomials. Furthermore, as a step towards proving  $\text{ACC}^0$  lower bounds for the majority function via our approach, we show that MAJORITY cannot be approximated by low-degree *symmetric* torus polynomials. We also pose several open problems related to our framework.

**2012 ACM Subject Classification** Theory of computation → Circuit complexity

**Keywords and phrases** Circuit complexity, ACC, lower bounds, polynomials

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2019.13

**Related Version** <https://arxiv.org/abs/1804.08176>

**Funding** Supported by NSF grant CCF-1614023.

**Acknowledgements** We thank Marco Carmosino for useful discussions regarding this work. We would also like to thank Eric Allender, Sivakanth Gopi, and anonymous referees for helpful feedback on earlier drafts of this work.

## 1 Introduction

A major goal of complexity theory is to prove Boolean circuit lower bounds, i.e. find explicit Boolean functions that cannot be computed by small size circuits of a given type. Over the years, three general approaches have been developed to achieve this.

---

<sup>1</sup> Part of this work was done when the author was visiting the University of California, San Diego. Research supported in part by Rutgers AAUP-AFT TA-GA Professional Development Fund.



The first approach is based on random restrictions. It applies to circuit classes in which functions simplify when most inputs are fixed to random values. Classic examples are the proofs by Håstad that  $AC^0$ , i.e. polynomial size circuit families of constant depth consisting of AND, OR, and NOT gates, cannot compute or approximate the PARITY function [6], and the shrinkage of De Morgan formulas (Boolean circuits consisting of AND, OR, and NOT gates whose underlying graph is a tree) under random restrictions [7]. However, random restrictions don't seem to be useful against more powerful circuit classes such as  $AC^0[\oplus]$  – the class of  $AC^0$  circuits equipped with PARITY gates.

The second approach is based on approximation by low-degree polynomials. Razborov [10] and Smolensky [12] used this approach to prove lower bounds for  $AC^0[\oplus] = AC^0[2]$ , and more generally for  $AC^0[p]$  for any prime  $p$  (This is the class of  $AC^0$  circuits that are allowed to have  $MOD_p$  gates<sup>2</sup>). This technique is based on showing that any function in the circuit class can be approximated by a low-degree polynomial over the finite field  $\mathbb{F}_p$ . Then, functions that do not admit such an approximation are provably outside the circuit class. A classic example here is that the MAJORITY function cannot be approximated by a low-degree polynomial over  $\mathbb{F}_p$ , and thus cannot be computed by  $AC^0[p]$ . However, this method also breaks down when considering more powerful circuit classes such as  $AC^0[6]$ , and more generally  $ACC^0$ , i.e.  $AC^0$  circuits with  $MOD_m$  gates where  $m$  is a composite that is not a prime power.

The third method involves designing nontrivial satisfiability algorithms and then using them along with classical tools from structural complexity theory (among other techniques and results) to prove circuit lower bounds against  $ACC^0$  for functions in high complexity classes such as NEXP. Williams [16] used this approach to prove that  $NEXP \not\subseteq ACC^0$ , and very recently, Williams and Murray [8] have extended this to show that  $NQP \not\subseteq ACC^0$ .

The goal of this paper is to focus on the second approach, namely the use of algebraic techniques, and to try and extend these techniques to prove lower bounds against  $ACC^0$ . We show that an extension of finite field polynomials, which we call *torus polynomials*, is a concrete candidate to achieve this. In particular, using a slightly stronger version of a result of Green et al. [5], we show that functions in  $ACC^0$  can be approximated<sup>3</sup> by low-degree torus polynomials. We remark that torus polynomials also generalize the class of *nonclassical polynomials* which arose in number theory and in higher order Fourier analysis [14], and are closely related to them.

This characterization of  $ACC^0$  using torus polynomials raises a host of questions on the approximation of Boolean functions by torus polynomials, the most remarkable being the problem of finding an explicit Boolean function that cannot be approximated by low-degree torus polynomials; an answer to this question would imply  $ACC^0$  lower bounds. In this paper, we take steps towards trying to resolve this question by initiating the study of approximation of Boolean functions by torus polynomials and proving some interesting results along the way. The motivation for our work is two-fold:

1. Given the slew of recent works exploring properties and applications of nonclassical polynomials [13, 14, 2, 3, 4], and the fact that torus polynomials are closely related to nonclassical polynomials, we believe that our characterization of  $ACC^0$  using torus polynomials might pave a way for new  $ACC^0$  lower bounds.
2. While the works of Williams [16] and Williams and Murray [8] are groundbreaking and prove highly nontrivial lower bounds against  $ACC^0$ , their proofs are not purely

<sup>2</sup> a  $MOD_p$  gate outputs 1 if and only if the sum of its inputs is congruent to a non-zero value modulo  $p$ .

<sup>3</sup> The notion of approximation that we use will be made explicit in Section 1.1.

combinatorial/algebraic, and it will be interesting to recover their results using purely algebraic/combinatorial techniques. We hope that our work will renew interest in this line of inquiry.

## 1.1 Torus polynomials

Let  $\mathbb{T} = \mathbb{R}/\mathbb{Z}$  denote the one-dimensional torus. A *torus polynomial* is simply a real polynomial restricted to the domain  $\{0, 1\}^n$  and evaluated modulo one<sup>4</sup>. Namely, a degree- $d$  torus polynomial  $P : \{0, 1\}^n \rightarrow \mathbb{T}$  is

$$P(x) = \sum_{S \subseteq [n], |S| \leq d} P_S \prod_{i \in S} x_i \pmod{1},$$

where  $P_S \in \mathbb{R}$ .

As it shall become evident later, torus polynomials extend finite field polynomials in that they provide a uniform way to capture computation of Boolean functions by polynomials over different finite fields – if a function can be computed by a low-degree polynomial over a finite field then it can be approximated by a low-degree torus polynomial. We will discuss this in detail in Section 2.

For  $z \in \mathbb{T}$ , let  $\iota(z)$  denote the unique representative of  $z$  in  $[-1/2, 1/2)$  (e.g.,  $\iota(0.4) = 0.4$  and  $\iota(0.7) = -0.3$ ). Then we can define its norm, denoted by  $|z \pmod{1}|$ , to be

$$|z \pmod{1}| = |\iota(z)|.$$

For  $F : \{0, 1\}^n \rightarrow \mathbb{T}$ , define

$$\|F \pmod{1}\|_\infty := \max_{x \in \{0, 1\}^n} |F(x) \pmod{1}|.$$

We embed Boolean functions as functions mapping into the torus by enforcing their output to be in  $\{0, 1/2\} \subset \mathbb{T}$  (This can be achieved by scaling the output of the function by  $1/2$ ). The following is the main definition of approximation that we consider:

► **Definition 1.** Let  $f : \{0, 1\}^n \rightarrow \{0, 1\}$  be a Boolean function. For  $\varepsilon > 0$ , a torus polynomial  $P : \{0, 1\}^n \rightarrow \mathbb{T}$  is said to  $\varepsilon$ -approximate  $f$  if

$$\left\| P - \frac{f}{2} \pmod{1} \right\|_\infty \leq \varepsilon.$$

Intuitively, a torus polynomial that approximates  $f$  takes a value “close” to 0 in the torus  $\mathbb{T}$  whenever  $f$  takes the value 0, and takes a value “close” to  $1/2$  in the torus whenever  $f$  takes the value 1.

We now introduce the notion of the *toroidal approximation degree* of a Boolean function.

► **Definition 2 (Toroidal approximation degree of Boolean functions).** Let  $f : \{0, 1\}^n \rightarrow \{0, 1\}$  be a Boolean function. For  $\varepsilon > 0$ , the toroidal  $\varepsilon$ -approximation degree of  $f$  is the minimal  $d \geq 0$ , for which there exists a torus polynomial  $P : \{0, 1\}^n \rightarrow \mathbb{T}$  of degree  $d$ , that satisfies

$$\left\| P - \frac{f}{2} \pmod{1} \right\|_\infty \leq \varepsilon.$$

We denote this by  $\overline{\text{deg}}_\varepsilon(f) = d$ .

<sup>4</sup> For  $x \in \mathbb{R}$ ,  $x$  modulo one, denoted by  $x \pmod{1}$ , is equal to the fractional part of  $x$  given by  $x - \lfloor x \rfloor$ , where  $\lfloor x \rfloor$  is the floor function. For example,  $2.6 \pmod{1}$  is 0.6, and  $-1.3 \pmod{1}$  is 0.7.

We illustrate in Section 2, in increasing generality, the power of torus polynomials. The most general result (Corollary 20) shows that if  $f$  can be computed by an  $\text{ACC}^0$  circuit then

$$\overline{\text{deg}}_\varepsilon(f) \leq \text{polylog}(n/\varepsilon).$$

The proof of this result uses a slightly stronger version of a result of Green et al. [5].

The above characterization paves way for a new approach to proving lower bounds against  $\text{ACC}^0$  for an explicit function, ideally in the class P. Concretely, we pose the following open problem.

► **Problem 3.** *Find an explicit function  $f : \{0,1\}^n \rightarrow \{0,1\}$  in P whose toroidal  $\varepsilon$ -approximation degree is  $\omega(\text{polylog}(n/\varepsilon))$ . By Corollary 20, it cannot be computed by  $\text{ACC}^0$  circuits.*

Williams [16] proved that  $\text{NEXP} \not\subseteq \text{ACC}^0$  via designing nontrivial satisfiability algorithms for  $\text{ACC}^0$ , and Williams and Murray [8] improved the approach to show that  $\text{NQP} \not\subseteq \text{ACC}^0$ . Thus, an intermediate goal towards resolving Problem 3 is to prove toroidal approximation lower bounds for functions  $f \in \text{NEXP}$  or  $f \in \text{NQP}$ .

A long-standing open problem in circuit complexity is to show that MAJORITY cannot be computed in  $\text{ACC}^0$ . Thus the following question is natural.

► **Problem 4.** *What is the toroidal  $\varepsilon$ -approximation degree of MAJORITY?*

How can one go about answering this question? We now turn to the setting of approximation of Boolean functions by real polynomials – which *prima facie* shares some similarities with our setting – for inspiration, highlighting the main differences between the two notions.

## 1.2 Comparison with real polynomials

Given a function  $f : \{0,1\}^n \rightarrow \{0,1\}$ , the real  $\varepsilon$ -approximation degree of  $f$ , denoted by  $\widetilde{\text{deg}}_\varepsilon(f)$ , is the minimal  $d$  such that there is a real polynomial  $P$  of degree  $d$  such that  $\|f - P\|_\infty \leq \varepsilon$  (this is the  $\ell_\infty$ -norm restricted to the domain  $\{0,1\}^n$ ). It is clear that  $\overline{\text{deg}}_\varepsilon(f) \leq \widetilde{\text{deg}}_\varepsilon(f)$ .

A beautiful result of Nisan and Szegedy [9] shows that the real  $\varepsilon$ -approximation degree of MAJORITY is  $\Omega(\sqrt{n})$  for  $\varepsilon < 1/2$ . Their proof proceeds in two stages: (i) showing that if a *symmetric* real polynomial  $\varepsilon$ -approximates MAJORITY then it must have degree  $\Omega(\sqrt{n})$ ; and (ii) that any polynomial that  $\varepsilon$ -approximates MAJORITY can be *symmetrized* and made into a symmetric polynomial with the same degree and approximation guarantee.

Attempting to follow the same strategy in the case of torus polynomials, we show in Corollary 23 in Section 3 that if one restricts attention to *symmetric* torus polynomials (namely, symmetric real polynomials evaluated modulo one), then the toroidal  $(1/20n)$ -approximation degree of MAJORITY is  $\Omega(\sqrt{n/\log n})$ .

Unfortunately, the aforementioned idea of symmetrization cannot be used in the setting of torus polynomials in a straightforward manner and so it's unclear how powerful non-symmetric torus polynomials are compared to their symmetric counterparts. We conjecture that they are not any better at approximating MAJORITY than symmetric torus polynomials:

► **Conjecture 5.** *The toroidal  $(1/20n)$ -approximation degree of MAJORITY is  $\Omega(\sqrt{n/\log n})$ .*

We remark that a positive answer to the above conjecture will give an algebraic proof that MAJORITY is not in  $\text{ACC}^0$ .



Let  $\Delta_w : \{0, 1\}^n \rightarrow \{0, 1\}$  denote the delta function which takes the value 1 on inputs of Hamming weight  $w$  and is 0 elsewhere. En route to proving the aforementioned lower bound for MAJORITY we also prove lower bounds for the delta functions in Theorem 21, showing that one needs symmetric torus polynomials of degree  $\Omega(\sqrt{n/\log n})$  in order to be able to  $(1/20n)$ -approximate the delta functions.

Somewhat surprisingly, for relatively large values of  $\varepsilon$ , the delta functions can be nontrivially  $\varepsilon$ -approximated by low-degree *symmetric* torus polynomials. In particular, we show in Lemma 24 in Section 4 that for every delta function there is a symmetric torus polynomial of degree  $\text{polylog}(n/\varepsilon)/\varepsilon$  that  $\varepsilon$ -approximates it, and thus

$$\overline{\text{deg}}_\varepsilon(\Delta_w) \leq \frac{\text{polylog}(n/\varepsilon)}{\varepsilon}.$$

This kind of dependence of the toroidal approximation degree on  $\varepsilon$  is quite interesting, and is unlike the case of real approximation – the real approximation degree of the delta functions is  $\Omega(\sqrt{n})$  for both small and large values of  $\varepsilon$ . In fact, for constant  $\varepsilon$ , this also shows a super-polynomial separation between real and toroidal approximation degree.

This also highlights other major differences between the real and the toroidal setting. Nisan and Szegedy [9] show that for every Boolean function the real approximation degree is polynomially related to the degree of exact representation by real polynomials. However, in the case of torus polynomials, this is not true: the delta functions require the degree to be  $\Omega(n)^5$  for exact representation whereas their toroidal  $1/3$ -approximation degree is  $O(\text{polylog}(n))$ .

An interesting property of real approximation is its amenability to amplification, namely the fact that, for any Boolean function  $f$  and  $\varepsilon < 1/3$ , given a polynomial  $p$  of degree  $d$  that  $1/3$ -approximates  $f$ , it can be transformed into a polynomial  $p'$  of degree  $d' = O(d \log(1/\varepsilon))$  that  $\varepsilon$ -approximates  $f$ . In other words,  $\overline{\text{deg}}_\varepsilon(f) \leq O(\overline{\text{deg}}_{1/3}(f) \log(1/\varepsilon))$ . It is not clear whether such a transformation is possible in the case of toroidal approximation. In the case of real approximation, the transformation is symmetry preserving, but, given the results for the delta functions discussed in the previous paragraphs, we should not expect this in the toroidal case. This motivates the following problem.

► **Problem 6.** *How is  $\overline{\text{deg}}_\varepsilon(f)$  related to  $\overline{\text{deg}}_{1/3}(f)$ ?*

### 1.3 Comparison with nonclassical polynomials

As mentioned before, torus polynomials generalize the class of nonclassical polynomials (this will be evident from the definition of nonclassical polynomials stated below). We remark that the results of this paper can be similarly phrased in terms of nonclassical polynomials instead of torus polynomials. This is because for the purpose of approximation of Boolean functions – which is the topic of this paper – torus polynomials and nonclassical polynomials are equivalent, as we shall see below. However, torus polynomials are simpler to describe (they are just real polynomials evaluated modulo 1) and more elegant (they are field independent), and hence we believe are a better choice for an algebraic model and for stating our results.

We now give the definition of nonclassical polynomials; here we provide what is known as the global definition of nonclassical polynomials over  $\{0, 1\}^n$ . For simplicity, we restrict our attention to nonclassical polynomials defined over  $\mathbb{F}_2^n$ , but note that the results generalize to nonclassical polynomials defined over  $\mathbb{F}_p^n$  for any constant prime  $p$ .

<sup>5</sup> To see this, note that the delta function  $\Delta_n(x)$  has a unique representation as a torus polynomial given by  $\Delta_n(x) = \frac{x_1 \cdots x_n}{2}$ .

► **Definition 7** (Nonclassical polynomials). A function  $Q : \{0, 1\}^n \rightarrow \mathbb{T}$  is a nonclassical polynomial (over  $\mathbb{F}_2$ ) of degree at most  $d$  if and only if it can be written as

$$Q(x) = \alpha + \sum_{\emptyset \subset S \subseteq [n]; k \geq 0; 0 < |S| + k \leq d} \frac{c_{S,k}}{2^{k+1}} \prod_{i \in S} x_i \pmod{1}$$

where  $c_{S,k} \in \{0, 1\}$  and  $\alpha \in \mathbb{T}$ .

The following simple claim shows that torus polynomials can be approximated by nonclassical polynomials.

► **Claim 8.** Let  $P : \{0, 1\}^n \rightarrow \mathbb{T}$  be a torus polynomial of degree at most  $d$  and let  $\varepsilon \in (0, 1)$ . Then there exists a nonclassical polynomial  $Q$  of degree at most  $O(d \log n + \log(1/\varepsilon))$  such that  $\|P - Q \pmod{1}\|_\infty \leq \varepsilon$ .

**Proof.** Suppose  $P(x) = \alpha + \sum_{\emptyset \subset S \subseteq [n], |S| \leq d} P_S \prod_{i \in S} x_i \pmod{1}$ . We can assume without loss of generality that  $P_S \in [0, 1)$  for all  $S$ . We approximate each  $P_S$  separately using dyadic rationals. Let  $P_S = 0.c_{S,0}c_{S,1}c_{S,2}\dots$ , where  $c_{S,i} \in \{0, 1\}$ , be its binary expansion. Let  $t \geq 1$  be a parameter that we will fix later, and note that

$$\left| P_S - \sum_{0 \leq k \leq t} \frac{c_{S,k}}{2^{k+1}} \right| \leq 2^{-t}.$$

Define the nonclassical polynomial

$$Q(x) = \alpha + \sum_{\emptyset \subset S \subseteq [n]; k \geq 0; 0 < |S| + k \leq t+d} \frac{c'_{S,k}}{2^{k+1}} \prod_{i \in S} x_i \pmod{1},$$

where  $c'_{S,k} = c_{S,k}$  for  $|S| \leq d, k \leq t$ , and is 0 otherwise. Then  $\deg(Q) \leq t + d$ , and

$$|P(x) - Q(x) \pmod{1}| \leq \binom{n}{\leq d} 2^{-t}$$

for all  $x \in \{0, 1\}^n$ . Choosing  $t = O(d \log n + \log(1/\varepsilon))$  completes the proof. ◀

Recall that our goal, motivated by proving ACC<sup>0</sup> lower bounds, is to find a Boolean function which cannot be  $1/\text{poly}(n)$ -approximated by a torus polynomial of degree  $\text{polylog}(n)$ . Given Claim 8, this is equivalent to the problem of finding a Boolean function which cannot be  $1/\text{poly}(n)$ -approximated by a nonclassical polynomial of degree  $\text{polylog}(n)$ . As we mentioned before, owing to the elegance and ease of description of torus polynomials relative to nonclassical polynomials, torus polynomials make for a more convenient choice in our setting.

## 1.4 Comparison with other notions of approximation

It's clear from our discussion in the previous section that torus polynomials are closely related to nonclassical polynomials, and so it's worthwhile to discuss two notions of approximation of Boolean functions by nonclassical polynomials that have been studied in the literature. The first deals with the *exact* computation of a Boolean function by a nonclassical polynomial on a nontrivial fraction of the domain [3]. For example, the work of Bhrushundi et al.[4] shows that any polynomial that computes MAJORITY correctly even on two-thirds of the points must have degree  $\Omega(\sqrt{n})$ . While many of these bounds for nonclassical polynomials should also hold for torus polynomials, we remark that they are not relevant to our setting since our notion of approximation (i.e., point-wise) is incomparable with the above notion.

The second notion is that of correlation with polynomials, which was studied, for example, by Bhowmick and Lovett[3]. Without getting into definitions here, we note that this notion of approximation is *weaker* than that of point-wise approximation<sup>6</sup>, and thus for the purpose of proving lower bounds for  $\text{ACC}^0$  it makes sense to work with only the latter. This also means that the upper bound results proved in the work of Bhowmick and Lovett (i.e., showing how certain Boolean functions can be approximated by low-degree nonclassical polynomials in the correlation sense) don't have any implications for our setting. Even their lower bound results, unfortunately, are not useful for us given that they only work against polynomials of degree  $\ll \log(n)$ , whereas we are dealing with polynomials of degree  $\text{polylog}(n)$ .

## 1.5 Natural proofs

An interesting line of inquiry motivated by the *natural proofs* framework of Razborov and Rudich [11] is whether the property of being inapproximable by low-degree torus polynomials is *natural*. It is not hard to see that it satisfies the *largeness* condition though it's unclear whether it is *constructive*, and so it will be interesting to investigate whether one can efficiently distinguish between Boolean functions which can be approximated by low-degree torus polynomials and a random Boolean function.

► **Problem 9.** *Given the truth table of a function  $f : \{0, 1\}^n \rightarrow \{0, 1\}$  and  $\varepsilon > 0$ , decide in polynomial time (in  $2^n$  and  $1/\varepsilon$ ) whether  $\text{deg}_\varepsilon(f) \leq \text{polylog}(n/\varepsilon)$ .*

### Paper organization

In Section 2, we prove toroidal approximation results for Boolean functions in bounded circuit classes such as  $\text{AC}^0[p]$  and  $\text{ACC}^0$ . In Section 3, we prove lower bounds against symmetric torus polynomials approximating the MAJORITY function and the delta functions. In Section 4, we show that symmetric torus polynomials have surprising power in approximating the delta functions when the error  $\varepsilon$  is not too small. We introduce definitions and notation along the way, as and when needed.

## 2 Approximation of circuit classes

In this section, we illustrate how the framework of approximation by torus polynomials captures computation of Boolean functions in various models of computation. We begin by showing that functions that are computable by low-degree polynomials over finite fields can be approximated by low-degree torus polynomials.

It might be instructive to keep in mind that, for the scope of the entire paper, whenever we consider polynomials, we restrict ourselves to only *multilinear* polynomials, i.e. polynomials in which the maximum degree of any variable is at most 1. Even if we encounter polynomials that do not adhere to this form during intermediate steps in certain proofs, we can always *multilinearize* the polynomials by making the degrees of all the variables equal to 1 wherever they appear. It suffices to consider multilinear polynomials because we always restrict the variables to the domain  $\{0, 1\}$ .

<sup>6</sup> By this we mean that if a function is point-wise approximated by a low-degree torus polynomial then it is also approximated by that polynomial in the correlation sense.

### 2.1 Polynomials over finite fields

Let  $\mathbb{F}_p$  be a prime finite field. We say a polynomial  $P(x) \in \mathbb{F}_p[x_1, \dots, x_n]$  computes a Boolean function  $f$  if

$$\forall x \in \{0, 1\}^n, f(x) = P(x).$$

Consider a function  $f$  which is computed by a low-degree polynomial over  $\mathbb{F}_p$ . We will now show that it can be approximated by a low-degree torus polynomial. We would require the following theorem on modulus-amplifying polynomials of Beigel and Tarui [1], following previous results of Toda [15] and Yao [17].

► **Lemma 10** (Beigel and Tarui [1]). *For every  $k \geq 1$ , there exists a univariate polynomial  $A_k : \mathbb{Z} \rightarrow \mathbb{Z}$  of degree  $2k - 1$  such that the following holds. For every  $m \geq 2$ ,*

- *If  $x \in \mathbb{Z}$  satisfies  $x \equiv 0 \pmod{m}$  then  $A_k(x) \equiv 0 \pmod{m^k}$ .*
- *If  $x \in \mathbb{Z}$  satisfies  $x \equiv 1 \pmod{m}$  then  $A_k(x) \equiv 1 \pmod{m^k}$ .*

► **Lemma 11.** *Let  $f : \{0, 1\}^n \rightarrow \{0, 1\}$ . Assume that  $f$  can be computed by a polynomial over  $\mathbb{F}_p$  of degree  $d$ . Then for every  $\varepsilon > 0$ ,*

$$\overline{\deg}_\varepsilon(f) \leq O(d \log(1/\varepsilon)).$$

**Proof.** Since  $f$  is computable by degree- $d$  polynomials over  $\mathbb{F}_p$ , there must be an integer polynomial  $F(x)$  (i.e., a polynomial with coefficients in  $\mathbb{Z}$ ) of degree  $d$  such that

$$F(x) \equiv f(x) \pmod{p} \quad \forall x \in \{0, 1\}^n.$$

Let  $k \geq 1$  be large enough so that  $1/p^k \leq \varepsilon$ . Let  $0 \leq q \leq p^k - 1$  be such that

$$\left| \frac{q}{p^k} - \frac{1}{2} \pmod{1} \right| \leq \varepsilon.$$

Define

$$G(x) = \frac{qA_k(F(x))}{p^k} \pmod{1}.$$

We claim that

$$\left| G(x) - \frac{f(x)}{2} \pmod{1} \right| \leq \varepsilon \tag{1}$$

for all  $x$ . To see this, fix  $x$ , and recall that  $F(x) \equiv f(x) \pmod{p}$ , which means that  $A_k(F(x)) \equiv f(x) \pmod{p^k}$ , and hence  $G(x) \equiv \frac{q}{p^k} f(x) \pmod{1}$ . (1) now follows from our choice of  $q$ .

Noting that the degree of  $G$  is  $(2k - 1)d \leq O(d \log(1/\varepsilon))$  completes the proof. ◀

We will later need the following simple variant of Lemma 11. Its proof is identical.

► **Lemma 12.** *Let  $f : \{0, 1\}^n \rightarrow \{0, 1\}$ . Assume that  $f$  can be computed by a polynomial over  $\mathbb{F}_p$  of degree  $d$ . Then for every  $\alpha \in [0, 1]$  and every  $\varepsilon > 0$ , there exists a torus polynomial  $P : \{0, 1\}^n \rightarrow \mathbb{T}$  of degree  $O(d \log(1/\varepsilon))$  such that*

$$\|P - \alpha f \pmod{1}\|_\infty \leq \varepsilon.$$

## 2.2 Circuit class $AC^0[p]$

Recall that, for a fixed prime  $p$ ,  $AC^0[p]$  is the class of functions computable by polynomial size circuits of constant depth, consisting of AND, OR, NOT, and  $MOD_p$  gates. Here a  $MOD_p$  gate is one that outputs 1 if and only if the sum of its inputs is congruent to a non-zero value modulo  $p$ .

Let  $f : \{0, 1\}^n \rightarrow \{0, 1\}$  be a function in  $AC^0[p]$ . We show that it can also be approximated by low-degree torus polynomials. The starting point is the classic result of Razborov [10] and Smolensky [12] which shows that  $AC^0[p]$  circuits can be approximated by random low-degree polynomials over  $\mathbb{F}_p$  in the following sense.

► **Theorem 13** (Razborov-Smolensky[10, 12]). *Let  $f : \{0, 1\}^n \rightarrow \{0, 1\}$  be computed by an  $AC^0[p]$  circuit. Then for every  $\varepsilon > 0$ , there exists a distribution  $\nu$  supported on polynomials  $F : \mathbb{F}_p^n \rightarrow \{0, 1\}$  of degree  $d = \text{polylog}(n/\varepsilon)$  such that*

$$\Pr_{P \sim \nu} [P(x) = f(x)] \geq 1 - \varepsilon \quad \forall x \in \{0, 1\}^n.$$

We can assume without loss of generality that all the polynomials in the support of the distribution  $\nu$  have range  $\{0, 1\}$ . This is because given an arbitrary polynomial  $P(x)$  over  $\mathbb{F}_p$  we can convert it into the polynomial  $P'(x) = (P(x))^{p-1}$  which has range  $\{0, 1\}$  by Fermat's little theorem. Note that the degree of  $P'$  is at most  $p$  times the degree of  $P$  which is not really a problem since  $p = O(1)$  for us.

We now show why torus polynomials approximate  $AC^0[p]$  functions.

► **Lemma 14.** *Let  $f : \{0, 1\}^n \rightarrow \{0, 1\}$ . Assume that there exists a distribution  $\nu$  supported on polynomials  $F : \mathbb{F}_p^n \rightarrow \{0, 1\}$  of degree  $d$  such that*

$$\Pr_{P \sim \nu} [P(x) = f(x)] \geq 1 - \varepsilon \quad \forall x \in \{0, 1\}^n.$$

Then

$$\overline{\text{deg}}_{3\varepsilon}(f) \leq O(d \log(n/\varepsilon)).$$

**Proof.** By standard Chernoff bounds, if we sample  $F_1, \dots, F_m \sim \nu$  independently for  $m = O(n/\varepsilon^2)$  then with high probability,

$$|\{i \in [m] : F_i(x) \neq f(x)\}| \leq 2\varepsilon m \quad \forall x \in \{0, 1\}^n.$$

Fix such a sample. Recall that  $F_i : \mathbb{F}_p^n \rightarrow \{0, 1\}$  are computed by degree  $d$  polynomials over  $\mathbb{F}_p$ . Next, apply Lemma 12 with  $\alpha = 1/2m$  and error  $\varepsilon/m$ . This gives us torus polynomials  $P_i : \{0, 1\}^n \rightarrow \mathbb{T}$  of degree  $O(d \log(m/\varepsilon))$  such that

$$\left| P_i(x) - \frac{1}{2m} F_i(x) \pmod{1} \right| \leq \frac{\varepsilon}{m} \quad \forall x \in \{0, 1\}^n.$$

Finally, take

$$P(x) = P_1(x) + \dots + P_m(x) \pmod{1}.$$

We claim that  $P(x)$  is a torus polynomial which  $3\varepsilon$ -approximates  $f(x)$ . To see this, fix  $x \in \{0, 1\}^n$ , and observe that

$$\left| P(x) - \frac{F_1(x) + \dots + F_m(x)}{2m} \pmod{1} \right| \leq \varepsilon$$

## 13:10 Torus Polynomials

and

$$\left| \frac{F_1(x) + \dots + F_m(x)}{2m} - \frac{f(x)}{2} \pmod{1} \right| \leq 2\varepsilon,$$

and so

$$\left| P(x) - \frac{f(x)}{2} \pmod{1} \right| \leq 3\varepsilon.$$

This means that

$$\overline{\deg}_{3\varepsilon}(f) \leq \deg(P) = \max\{\deg(P_i) : i \in [m]\} = O(d \log(m/\varepsilon)) = O(d \log(n/\varepsilon)). \quad \blacktriangleleft$$

► **Corollary 15.** *Let  $f : \{0, 1\}^n \rightarrow \{0, 1\}$  be a function in  $AC^0[p]$ . Then for every  $\varepsilon > 0$ ,*

$$\overline{\deg}_\varepsilon(f) \leq \text{polylog}(n/\varepsilon).$$

An interesting question that is motivated by the above results is whether we can have a mini-max type theorem for torus polynomials. Lemma 14 gives such a theorem in a very limited regime. The following is an attempt to generalize this.

► **Problem 16.** *Let  $f : \{0, 1\}^n \rightarrow \{0, 1\}$ . Assume that for any distribution  $\nu$  over  $\{0, 1\}^n$ , there exists a low-degree torus polynomial  $P_\nu : \{0, 1\}^n \rightarrow \mathbb{T}$  such that*

$$\mathbb{E}_{x \sim \nu} \left[ \left| P_\nu(x) - \frac{f(x)}{2} \pmod{1} \right| \right] \leq \varepsilon.$$

*Does that imply that the toroidal approximation degree of  $f$  is small? That is, does there exist a single low-degree torus polynomial which approximates  $f$  on all inputs?*

It might also be useful to assume the stronger assumption that for any distribution  $\nu$  over  $\{0, 1\}^n$  and any  $\alpha \in [0, 1]$  there exists a torus polynomial  $P_{\nu, \alpha} : \{0, 1\}^n \rightarrow \mathbb{T}$  of degree  $d$  such that

$$\mathbb{E}_{x \sim \nu} [|P_{\nu, \alpha}(x) - \alpha f(x) \pmod{1}|] \leq \varepsilon.$$

This is also related to the following problem.

► **Problem 17.** *Let  $f : \{0, 1\}^n \rightarrow \{0, 1\}$ . For any  $\alpha \in [0, 1]$  and  $\varepsilon > 0$  define  $d(\alpha, \varepsilon)$  to be the minimal degree of a torus polynomial  $P : \{0, 1\}^n \rightarrow \mathbb{T}$  such that*

$$\|P - \alpha f \pmod{1}\|_\infty \leq \varepsilon.$$

*What is the behavior of  $d(\alpha, \varepsilon)$  as a function of  $\alpha$  and of  $\varepsilon$ ? Specifically,*

- *Can we bound  $\max_\alpha d(\alpha, \varepsilon)$  in terms of  $d(1/2, \varepsilon)$ ?*
- *Can we bound  $\max_\alpha d(\alpha, \varepsilon)$  in terms of  $\max_\alpha d(\alpha, 0.1)$ ?*

### 2.3 Circuit class $ACC^0$

We now turn our attention to  $ACC^0$  functions and show that they too can be approximated by low-degree torus polynomials. Recall that a function is in  $ACC^0$  if it can be computed by polynomial size circuits of constant depth with AND, OR, NOT, and  $MOD_m$  gates where  $m$  may be composite.

Our starting point is the following result of Green et al. [5] which extends previous results of [17, 1].

► **Theorem 18** (Green et al. [5]). *Let  $f : \{0, 1\}^n \rightarrow \{0, 1\}$  be computable by  $\text{ACC}^0$  circuits of depth  $\ell$  and size  $\text{poly}(n)$ . Then for any  $e \geq 1$  there exists an integer polynomial  $F(x)$  of degree  $d = e^{O(\ell)} \log^{O(\ell^2)} n$  which satisfies the following: there is some  $k \geq 1$  such that*

$$\forall x \in \{0, 1\}^n, \quad F(x) = f(x)2^k + E(x) \pmod{2^{k+e}}$$

for some error  $E(x) \leq 2^{k-1}$ .

Note that the above theorem states that the  $k^{\text{th}}$  bit of  $F(x)$  in binary always equals to  $f(x)$  and that it's padded with  $e - 1$  zeros to its left, i.e the  $(k + 1)^{\text{th}}$ ,  $(k + 2)^{\text{th}}$ ,  $\dots$ ,  $(k + e - 1)^{\text{th}}$  bits are all guaranteed to be equal to 0. It turns out that, implicit in their work, is the following slightly stronger version of the above result which lets us pad zeros on both sides of the output bit (i.e., the  $k^{\text{th}}$  bit).

► **Theorem 19** (Implicit in Green et al. [5]). *Let  $f : \{0, 1\}^n \rightarrow \{0, 1\}$  be computable by  $\text{ACC}^0$  circuits of depth  $\ell$  and size  $\text{poly}(n)$ . Then for any  $e \geq 1$  there exists an integer polynomial  $F(x)$  of degree  $d = e^{O(\ell)} \log^{O(\ell^2)} n$  which satisfies the following: there is some  $k \geq e$  such that*

$$\forall x \in \{0, 1\}^n, \quad F(x) = f(x)2^k + E(x) \pmod{2^{k+e}}$$

for some error  $E(x) \leq 2^{k-e}$ .

Note the difference between the statements of Theorem 18 and Theorem 19: while the former upper-bounds the error  $E(x)$  by  $2^{k-1}$  the latter bounds it by  $2^{k-e}$ , thus padding the output bit with  $e - 1$  zeros on both the sides.

Since the proof of Theorem 19 is essentially the same as that of Theorem 18 with some very minor tweaks, we choose to omit it here. We now show how to use Theorem 19 to prove that low-degree torus polynomials approximate functions in  $\text{ACC}^0$ .

► **Corollary 20.** *Let  $f : \{0, 1\}^n \rightarrow \{0, 1\}$  be a function in  $\text{ACC}^0$ . Then for every  $\varepsilon > 0$ , there is a torus polynomial of degree  $\text{polylog}(n/\varepsilon)$  that  $\varepsilon$ -approximates  $f$ . In other words,*

$$\overline{\text{deg}}_\varepsilon(f) \leq \text{polylog}(n/\varepsilon).$$

**Proof.** Let us assume that  $f$  is computable by  $\text{ACC}^0$  circuits of size  $\text{poly}(n)$  and depth  $\ell$ . Recall that, by definition of  $\text{ACC}^0$ ,  $\ell = O(1)$ . Let  $F(x)$  be the polynomial obtained by applying Theorem 19 to  $f$  with  $e = \log(1/\varepsilon)$  such that for some  $k \geq e$

$$\forall x \in \{0, 1\}^n, \quad F(x) = f(x)2^k + E(x) \pmod{2^{k+e}}.$$

The degree of  $F(x)$  is  $d = e^{O(\ell)} \log^{O(\ell^2)} n = \text{polylog}(n/\varepsilon)$ . Define the following torus polynomial

$$P(x) = \frac{F(x)}{2^{k+1}} \pmod{1}.$$

Clearly  $\text{deg}(P) = d$ . For  $i \geq 0$ , let  $F_i(x)$  denote the  $i^{\text{th}}$  bit of  $F(x)$ . Then, by the definition of  $F$ ,

$$\frac{F(x)}{2^{k+1}} \pmod{1} = \sum_{i=0}^k 2^{i-k-1} F_i(x) \pmod{1} = \frac{f(x)}{2} + \sum_{i=0}^{k-e} 2^{i-k-1} F_i(x) \pmod{1}.$$

As  $F_i(x) \in \{0, 1\}$  for all  $i$ , we can bound

$$\left| P(x) - \frac{f(x)}{2} \pmod{1} \right| \leq 2^{-e} \leq \varepsilon \quad \forall x \in \{0, 1\}^n. \quad \blacktriangleleft$$

### 3 Lower bound for symmetric torus polynomials

In this section we prove a lower bound on the degree of *symmetric* torus polynomials that approximate MAJORITY. It will be instructive to think of symmetric torus polynomials as symmetric real polynomials evaluated modulo one. We start by examining the question for delta functions.

For  $x \in \{0, 1\}^n$ , let  $|x| = \sum x_i$  denote its Hamming weight. The delta function

$$\Delta_w : \{0, 1\}^n \rightarrow \{0, 1\},$$

for  $0 \leq w \leq n$ , is defined as

$$\Delta_w(x) = \begin{cases} 1 & |x| = w \\ 0 & \text{otherwise} \end{cases}.$$

► **Lemma 21.** *Let  $n, d$  be positive integers such that for every  $0 \leq w \leq n$  there exists a symmetric torus polynomial  $Q_w : \{0, 1\}^n \rightarrow \mathbb{T}$  of degree  $d$  that  $\frac{1}{20n}$ -approximates  $\Delta_w(x)$ . Then  $d = \Omega\left(\sqrt{\frac{n}{\log n}}\right)$ .*

**Proof.** Let  $\text{Sym}(n)$  denote the set of symmetric Boolean functions in  $n$  variables and let  $\text{SymPoly}_{d,k}(n)$  denote the set of symmetric torus polynomials in  $n$  variables of degree  $d$  whose coefficients are of the form  $q/2^k$  for  $q \in \{-(2^k - 1), \dots, 0, \dots, 2^k - 1\}$ .

Let  $f$  be an arbitrary function in  $\text{Sym}(n)$ . Abusing notation, we let  $f^{-1}(1)$  denote the set of weights of the layers of the Hamming cube where  $f$  takes value 1. Now define the torus polynomial  $Q_f$  as

$$Q_f(x) = \sum_{i \in f^{-1}(1)} Q_i(x) \pmod{1}.$$

It follows that  $Q_f$  is a symmetric torus polynomial of degree  $d$  that  $\frac{1}{20}$ -approximates  $f$ . Since  $Q_f$  is a symmetric torus polynomial, namely a symmetric real polynomial modulo one, it may be written without loss of generality as

$$Q_f(x) = \sum_{j=0}^d c_j \left(\sum x_i\right)^j \pmod{1},$$

where  $c_j \in [0, 1)$ . Let  $k \geq 0$  be an integer whose value we will fix later. For  $0 \leq j \leq d$ , let  $q_j \in \{-(2^k - 1), \dots, 0, \dots, 2^k - 1\}$  be such that

$$\left| \frac{q_j}{2^k} - c_j \right| \leq \frac{1}{2^k},$$

and define  $Q'_f$  to be the polynomial

$$Q'_f(x) = \sum_{j=0}^d \frac{q_j}{2^k} \cdot \left(\sum x_i\right)^j \pmod{1}.$$

Observe that for every  $x \in \{0, 1\}^n$ ,

$$|Q_f(x) - Q'_f(x) \pmod{1}| \leq \sum_{j=0}^d \left| \frac{q_j}{2^k} - c_j \right| \cdot |x|^j \leq \frac{(d+1) \cdot n^d}{2^k}.$$



If  $k$  is such that  $\frac{(d+1) \cdot n^d}{2^k} \leq \frac{1}{20}$  then

$$\|Q_f - Q'_f \pmod{1}\|_\infty \leq \frac{1}{20},$$

and so

$$\left\| \frac{f}{2} - Q'_f \pmod{1} \right\|_\infty \leq \left\| \frac{f}{2} - Q_f \pmod{1} \right\|_\infty + \|Q_f - Q'_f \pmod{1}\|_\infty \leq \frac{1}{10}.$$

Note that we can choose  $k = O(d \log n)$  while still satisfying the required condition on  $k$ .

So far we have shown that for every  $f \in \text{Sym}(n)$  there is a polynomial  $Q_f \in \text{SymPoly}_{d,k}(n)$  that  $1/10$ -approximates  $f$  where  $k = O(d \log n)$ . In the other direction, one can easily verify that every polynomial in  $\text{SymPoly}_{d,k}(n)$  can  $1/10$ -approximate *at most* one function in  $\text{Sym}(n)$ . This implies that

$$|\text{SymPoly}_{d,k}(n)| \geq |\text{Sym}(n)|.$$

Plugging in  $|\text{SymPoly}_{d,k}(n)| = 2^{(k+1)(d+1)}$  and  $|\text{Sym}(n)| = 2^n$ , and using  $k = O(d \log n)$ , yields the bound  $d = \Omega\left(\sqrt{\frac{n}{\log n}}\right)$ .  $\blacktriangleleft$

Before we proceed, we formally define MAJORITY on  $n$  bits, denoted by  $\text{Maj}_n(x)$ , as

$$\text{Maj}_n(x) = \begin{cases} 1 & |x| \geq \frac{n}{2} \\ 0 & \text{otherwise} \end{cases}.$$

► **Lemma 22.** *If there is a symmetric torus polynomial of degree  $o\left(\sqrt{\frac{n}{\log n}}\right)$  that  $\frac{1}{20n}$ -approximates  $\text{Maj}_n(x)$ , then for every  $0 \leq w \leq n$  there is a symmetric torus polynomial of degree  $o\left(\sqrt{\frac{n}{\log n}}\right)$  that  $\frac{1}{20n}$ -approximates  $\Delta_w(x)$ .*

**Proof.** Fix  $w$ . Let  $\Delta_{\geq w}(x)$  denote the function that takes value 1 iff  $|x| \geq w$ . Then we can write

$$\Delta_{\geq w}(x_1, \dots, x_n) = \text{Maj}_{2n+1}(x_1, \dots, x_n, c_1, \dots, c_{n+1}), \tag{2}$$

where  $c \in \{0, 1\}^{n+1}$  is the string whose first  $n - w + 1$  bits are set to 1 and the rest of the bits are set to 0. Let  $Q(x_1, \dots, x_{2n+1})$  be the symmetric torus polynomial in  $2n + 1$  variables that  $\frac{1}{20(2n+1)}$ -approximates  $\text{Maj}_{2n+1}(x)$ . Let  $Q_{\geq w}(x_1, \dots, x_n)$  be the torus polynomial defined as

$$Q_{\geq w}(x_1, \dots, x_n) = Q(x_1, \dots, x_n, c_1, \dots, c_{n+1}),$$

where  $c \in \{0, 1\}^{n+1}$  is as defined above. It follows from (2) that  $Q_{\geq w}(x_1, \dots, x_n)$   $\frac{1}{40n}$ -approximates  $\Delta_w(x_1, \dots, x_n)$ . Furthermore,

$$\text{deg}(Q_{\geq w}) = o\left(\sqrt{\frac{n}{\log n}}\right).$$

Similarly, we can obtain a symmetric torus polynomial  $Q_{\geq w+1}$  that  $\frac{1}{40n}$ -approximates  $\Delta_{\geq w+1}(x_1, \dots, x_n)$  such that

$$\text{deg}(Q_{\geq w+1}) = o\left(\sqrt{\frac{n}{\log n}}\right).$$

## 13:14 Torus Polynomials

Note that

$$\frac{\Delta_w(x)}{2} \pmod{1} = \left( \frac{\Delta_{\geq w}(x)}{2} - \frac{\Delta_{\geq w+1}(x)}{2} \right) \pmod{1}.$$

Defining  $Q_w(x) = Q_{\geq w}(x) - Q_{\geq w+1}(x) \pmod{1}$ , it follows that

$$\left\| \frac{\Delta_w(x)}{2} - Q_w(x) \pmod{1} \right\|_{\infty} \leq \frac{1}{20n}.$$

This completes the proof. ◀

The main result of this section now follows from Theorem 21 and Lemma 22:

► **Corollary 23.** *Any symmetric torus polynomial of degree  $d$  that  $\frac{1}{20n}$ -approximates  $\text{Maj}_n(x)$  must satisfy  $d = \Omega\left(\sqrt{\frac{n}{\log n}}\right)$ .*

### 4 Upper bound for delta functions

In this section, we prove the somewhat surprising result that if the approximation parameter  $\varepsilon > 0$  is not too small (say,  $\varepsilon$  is a small constant), then the delta function  $\Delta_w$  can be nontrivially approximated by *symmetric* low-degree torus polynomials.

► **Lemma 24.** *For every  $0 \leq w \leq n$  and  $\varepsilon > 0$ , there is a symmetric torus polynomial of degree  $\frac{\text{polylog}(n/\varepsilon)}{\varepsilon}$  that  $\varepsilon$ -approximates  $\Delta_w(x)$ , and thus*

$$\overline{\text{deg}}_{\varepsilon}(\Delta_w) \leq \frac{\text{polylog}(n/\varepsilon)}{\varepsilon}.$$

**Proof.** For any prime  $p \geq 2$ , let  $f_p : \{0, 1\}^n \rightarrow \{0, 1\}$  denote the function

$$f_p(x) = \begin{cases} 1 & |x| \equiv w \pmod{p} \\ 0 & \text{otherwise} \end{cases}.$$

It is computed by the  $\mathbb{F}_p$ -polynomial of degree  $p - 1$

$$f_p(x) = 1 - \left( \sum x_i - w \right)^{p-1} \pmod{p}.$$

Let  $P = \{p_1, \dots, p_t\}$  be the first  $t$  primes, for  $t$  to be chosen later. Applying Lemma 12 with  $\alpha = 1/2t$  and error  $\varepsilon/2t$ , for each  $p \in P$  we obtain a torus polynomial  $Q_p : \{0, 1\} \rightarrow \mathbb{T}$  of degree  $O(p \log(t/\varepsilon))$  such that

$$\left\| Q_p - \frac{1}{2t} f_p \pmod{1} \right\|_{\infty} \leq \frac{\varepsilon}{2t}.$$

Define

$$Q(x) = \sum_{p \in P} Q_p(x) \pmod{1}.$$

We claim that  $Q$  is a symmetric torus polynomial that  $\varepsilon$ -approximates  $\Delta_w$ .

Consider first  $x \in \{0, 1\}^n$  with  $|x| = w$ . In this case, for each  $p \in P$  we have  $f_p(x) = 1$ ,  $|Q_p(x) - \frac{1}{2t} \pmod{1}| \leq \varepsilon/2t$  and hence

$$\left| Q(x) - \frac{1}{2} \pmod{1} \right| \leq \varepsilon/2.$$

Next, assume that  $|x| \neq w$ . Then  $f_p(x) = 1$  only if  $p$  divides  $|x| - w$ . As there are at most  $\log n$  such primes, we have that

$$|Q(x) \pmod{1}| \leq \frac{\varepsilon}{2} + \frac{\log n}{t}.$$

To conclude we choose  $t = O(\log(n)/\varepsilon)$ . The largest prime in  $P$  has size  $O(t \log t)$  which means that

$$\overline{\deg}_\varepsilon(f) \leq \deg(Q) = \max\{\deg(Q_p) : p \in P\} \leq O(t \log t \cdot \log(t/\varepsilon)) = \frac{\text{polylog}(n/\varepsilon)}{\varepsilon}.$$

To see why  $Q$  is symmetric, observe that Lemma 12 preserves symmetry.  $\blacktriangleleft$

---

## References

- 1 Richard Beigel and Jun Tarui. On ACC (circuit complexity). In *Proceedings of the 32nd Annual Symposium on Foundations of Computer Science*, FOCS 1991, pages 783–792. IEEE, 1991.
- 2 Arnab Bhattacharyya, Eldar Fischer, Hamed Hatami, Pooya Hatami, and Shachar Lovett. Every Locally Characterized Affine-invariant Property is Testable. In *Proceedings of the Forty-fifth Annual ACM Symposium on Theory of Computing*, STOC 2013, pages 429–436. ACM, 2013.
- 3 Abhishek Bhowmick and Shachar Lovett. Nonclassical polynomials as a barrier to polynomial lower bounds. In *Proceedings of the 30th Conference on Computational Complexity*, CCC 2015, pages 72–87. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2015.
- 4 Abhishek Bhruhundi, Prahladh Harsha, and Srikanth Srinivasan. On Polynomial Approximations Over  $\mathbb{Z}/2^k\mathbb{Z}$ . In *Proceedings of the 34th Symposium on Theoretical Aspects of Computer Science*, STACS 2017, pages 12:1–12:12, 2017.
- 5 Frederic Green, Johannes Kobler, and Jacobo Toran. The power of the middle bit. In *Proceedings of the 7th Annual Structure in Complexity Theory Conference, 1992*, pages 111–117. IEEE, 1992.
- 6 Johan Håstad. *Computational Limitations of Small-depth Circuits*. MIT Press, Cambridge, MA, USA, 1987.
- 7 Johan Håstad. The shrinkage exponent of De Morgan formulas is 2. *SIAM Journal on Computing*, 27(1):48–64, 1998.
- 8 Cody Murray and Ryan Williams. Circuit Lower Bounds for Nondeterministic Quasipolytime: An Easy Witness Lemma for NP and NQP. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing*, STOC 2018, pages 890–901. ACM, 2018.
- 9 Noam Nisan and Mario Szegedy. On the Degree of Boolean Functions As Real Polynomials. In *Proceedings of the Twenty-fourth Annual ACM Symposium on Theory of Computing*, STOC 1992, pages 462–467. ACM, 1992.
- 10 Alexander A Razborov. Lower bounds for the size of circuits of bounded depth with basis  $\{\wedge, \oplus\}$ . *Math. notes of the Academy of Sciences of the USSR*, 41(4):333–338, 1987.
- 11 Alexander A Razborov and Steven Rudich. Natural Proofs. *Journal of Computer and System Sciences*, 55(1):24–35, 1997.
- 12 Roman Smolensky. Algebraic methods in the theory of lower bounds for Boolean circuit complexity. In *Proceedings of the 19th Annual ACM Symposium on Theory of computing*, STOC 1987, pages 77–82. ACM, 1987.
- 13 Terence Tao. Some notes on “non-classical” polynomials in finite characteristic, 2008.
- 14 Terence Tao and Tamar Ziegler. The inverse conjecture for the Gowers norm over finite fields in low characteristic. *Annals of Combinatorics*, 16(1):121–188, 2012.

## 13:16 Torus Polynomials

- 15 Seinosuke Toda. PP is as hard as the polynomial-time hierarchy. *SIAM Journal on Computing*, 20(5):865–877, 1991.
- 16 Ryan Williams. Nonuniform ACC circuit lower bounds. *Journal of the ACM (JACM)*, 61(1):2, 2014.
- 17 Andrew Chi-Chih Yao. Separating the polynomial-time hierarchy by oracles. In *Proceedings of the 26th Annual Symposium on Foundations of Computer Science*, FOCS 1985, pages 1–10. IEEE, 1985.

# Almost Envy-Free Allocations with Connected Bundles

**Vittorio Bilò**

University of Salento, Lecce, Italy

**Ioannis Caragiannis**

University of Patras, Rion-Patras, Greece

**Michele Flammini**

Gran Sasso Science Institute and University of L'Aquila, L'Aquila, Italy

**Ayumi Igarashi**

Kyushu University, Fukuoka, Japan

**Gianpiero Monaco**

University of L'Aquila, L'Aquila, Italy

**Dominik Peters**

University of Oxford, Oxford, U.K.

**Cosimo Vinci**

University of L'Aquila, L'Aquila, Italy

**William S. Zwicker**

Union College, Schenectady, USA

---

## Abstract

We study the existence of allocations of indivisible goods that are envy-free up to one good (EF1), under the additional constraint that each bundle needs to be connected in an underlying item graph  $G$ . When the items are arranged in a path, we show that EF1 allocations are guaranteed to exist for arbitrary monotonic utility functions over bundles, provided that either there are at most four agents, or there are any number of agents but they all have identical utility functions. Our existence proofs are based on classical arguments from the divisible cake-cutting setting, and involve discrete analogues of cut-and-choose, of Stromquist's moving-knife protocol, and of the Su-Simmons argument based on Sperner's lemma. Sperner's lemma can also be used to show that on a path, an EF2 allocation exists for any number of agents. Except for the results using Sperner's lemma, all of our procedures can be implemented by efficient algorithms. Our positive results for paths imply the existence of connected EF1 or EF2 allocations whenever  $G$  is traceable, i.e., contains a Hamiltonian path. For the case of two agents, we completely characterize the class of graphs  $G$  that guarantee the existence of EF1 allocations as the class of graphs whose biconnected components are arranged in a path. This class is strictly larger than the class of traceable graphs; one can check in linear time whether a graph belongs to this class, and if so return an EF1 allocation.

**2012 ACM Subject Classification** Theory of computation → Algorithmic game theory, Mathematics of computing → Combinatoric problems, Mathematics of computing → Graph theory

**Keywords and phrases** Envy-free Division, Cake-cutting, Resource Allocation, Algorithmic Game Theory

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2019.14



© Vittorio Bilò, Ioannis Caragiannis, Michele Flammini, Ayumi Igarashi, Gianpiero Monaco, Dominik Peters, Cosimo Vinci, and William S. Zwicker;  
licensed under Creative Commons License CC-BY

10th Innovations in Theoretical Computer Science (ITCS 2019).

Editor: Avrim Blum; Article No. 14; pp. 14:1–14:21



Leibniz International Proceedings in Informatics

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

**Related Version** [5], <https://arxiv.org/abs/1808.09406> with omitted proofs, and additional results

**Funding** D. Peters is supported by ERC grant 639945 (ACCORD). A. Igarashi is supported by KAKENHI (Grant-in-Aid for JSPS Fellows, No. 18J00997) Japan. While working on this paper, W. S. Zwicker was supported by the Oliver Smithies Visiting Fellowship at Balliol College, Oxford.

## 1 Introduction

A famous literature considers the problem of *cake-cutting* [10, 25, 24]. There, a divisible heterogeneous resource (a *cake*, usually formalized as the interval  $[0, 1]$ ) needs to be divided among  $n$  agents. Each agent has a valuation function over subsets of the cake, usually formalized as an atomless measure over  $[0, 1]$ . The aim is to partition the cake into  $n$  pieces, and allocate each piece to one agent, in a “fair” way. By fair, we will mean that the allocation is *envy-free*: no agent thinks that another agent’s piece is more valuable than her own.

When there are two agents, the classic procedure of cut-and-choose can produce an envy-free division: a knife is moved from left to right, until an agent shouts to indicate that she thinks the pieces to either side are equally valuable. The other agent then picks one of the pieces, leaving the remainder for the shouter. As is easy to see, the result is an envy-free allocation. For three or more agents, finding an envy-free division has turned out to be much trickier. An early result by Dubins and Spanier [15] used Lyapunov’s Theorem and measure-theoretic techniques to show, non-constructively, that an envy-free allocation always exists. However, as Stromquist [28] memorably writes, “their result depends on a liberal definition of a “piece” of cake, in which the possible pieces form an entire  $\sigma$ -algebra of subsets. A player who only hopes for a modest interval of cake may be presented instead with a countable union of crumbs.” In many applications of resource allocation (such as land division, or the allocation of time slots), agents have little use for a severely disconnected piece of cake.

Stromquist [28] himself offered a solution, and gave a new non-constructive argument (using topology) which proved that there always exists an envy-free division of the cake into *intervals*. Forest Simmons later observed that the proof could be simplified by using Sperner’s lemma, and this technique was subsequently presented in a paper by Su [29]. For the three-agent case, Stromquist [28] also presented an appealing moving-knife procedure that more directly yields a connected envy-free allocation. For  $n \geq 4$  agents, no explicit procedures are known to produce a connected envy-free allocation (i.e., an allocation where the cake is cut in exactly  $n - 1$  places). However, for  $n = 4$ , several moving-knife procedures exist that only need few cuts; for example, the Brams–Taylor–Zwicker [11] procedure requires 11 cuts, and a protocol of Barnabel and Brams [3] requires 5 cuts.

In many applications, the resources to be allocated are not infinitely divisible, and we face the problem of allocating *indivisible goods*. Most of the literature on indivisible goods has not assumed any kind of structure on the item space, in contrast to the rich structure of the interval  $[0, 1]$  in cake-cutting. Thus, there has been little attention on minimizing the number of “cuts” required in an allocation. However, when the items have a spatial or temporal structure, this consideration is important.

In this paper, we study the allocation of items that are arranged on a *path* or other structure, and impose the requirement that only *connected* subsets of items may be allocated to the agents. Formally, we work in the model of [9], who assume that the items form the vertex set of a graph  $G$ ; a bundle is connected if it induces a connected subgraph of  $G$ . In

their paper, it became apparent that techniques from cake-cutting can be usefully ported to achieve good allocations in the indivisible case. For example, moving-knife procedures that achieve proportionality in cake-cutting have analogues that produce allocations that satisfy the *maximin share guarantee* [12].<sup>1</sup>

Do envy-free procedures for cake-cutting also translate to the indivisible case? Of course, in general, it is impossible to achieve envy-freeness with indivisibilities (consider two agents and a single desirable item), but we can look for approximations. A relaxation of envy-freeness that has been very influential recently is envy-freeness *up to one good* (EF1), introduced by Budish [12]. It requires that an agent's envy towards another bundle vanishes if we remove some item from the envied bundle. In the setting without connectivity constraints and with additive valuations, the maximum Nash welfare solution satisfies EF1, as does a simple round-robin procedure [13]. The well-known envy-graph algorithm [21] also guarantees EF1. However, none of these procedures respects connectivity constraints.

When items are arranged on a path, we prove that connected EF1 allocations exist when there are two, three, or four agents. As was necessary in cake-cutting, we use successively more complicated tools to establish these existence results. For two agents, there is a discrete analogue of cut-and-choose that satisfies EF1. In that procedure, a knife moves across the path, and an agent shouts when the knife reaches what we call a *lumpy tie*, that is when the bundles to either side of the knife have equal value *up to one item*. For three agents, we design an algorithm mirroring Stromquist's moving-knife procedure which guarantees EF1. For four agents, we show that Sperner's lemma can be used to prove that an EF1 allocation exists, via a technique inspired by the Simmons–Su approach, and an appropriately triangulated simplex of connected partitions of the path. For five or more agents, we were not able to establish the existence of EF1 allocations on a path, but we can show (again via Sperner's lemma) that EF2 allocations exist, strengthening a prior result of Suksompong [30]. We also show that if all agents have the same valuation function over bundles, then an egalitarian-welfare-optimal allocation, after suitably reallocating some items, is EF1.

These existence results require only that agents' valuations are monotonic (they need not be additive), and in addition ensure that the constructed allocation satisfies the maximin share guarantee (see [5]). Moreover, the fairness guarantee of our algorithms is slightly stronger than the standard notion of EF1: in the returned allocations, envy can be avoided by removing just an *outer* item – one whose removal leaves the envied bundle connected. Computationally speaking, all our existence results are immediately useful, since an example of an EF1 allocation can be found by iterating through all  $O(m^n)$  connected allocation (this stands in contrast to cake-cutting where we cannot iterate through all possibilities). While we know of no faster algorithms to obtain an EF1 allocation in the cases where we appeal to Sperner's lemma, our other procedures can all be implemented efficiently.

In simultaneous and independent work, Oh et al. [23] designed protocols to find EF1 allocations in the setting without connectivity constraints, aiming for low *query complexity*. They found that adapting cake-cutting protocols to the setting of indivisible items arranged on a path is an especially potent way to achieve low query complexity. This led them to also study a discrete version of the cut-and-choose protocol which achieves connected EF1 allocations for two agents, and they found an alternative proof that an EF1 allocation on a path always exists with identical valuations. They also present a discrete analogue of the Selfridge–Conway procedure which, for three agents with additive valuations, produces an

---

<sup>1</sup> Another paper by Suksompong [30] works in the same model, and also found that procedures for proportionality and other concepts can be applied to the indivisible setting.

allocation of a path into bundles that have a constant number of connected components. However, they do not study connected allocations on graphs that are not paths, and they do not consider the case of (non-identical) general valuations with more than two agents.

A recurring theme in our algorithms is the specific way that the moving knives from cake-cutting are rendered in the discrete setting. While one might expect knives to be placed over the edges of the path, and “move” from edge to edge, we find that this movement is too “fast” to ensure EF1 (see also footnote 4). Instead, our knives alternate between hovering over edges and items. When a knife hovers over an item, we imagine the knife’s blade to be “thick”: the knife *covers* the item, and agents then pretend that the covered item does not exist. These intermediate steps are useful, since they can tell us that envy will vanish if we hide an item from a bundle.

What about graphs  $G$  other than paths? Our positive results for paths immediately generalize to traceable graphs (those that contain a Hamiltonian path), since we can run the algorithms pretending that the graph only consists of the Hamiltonian path. For the two-agent case, we completely characterize the class of graphs that guarantee the existence of EF1 allocations: Our discrete cut-and-choose protocol can be shown to work on all graphs  $G$  that admit a *bipolar numbering*, which exists if and only if the biconnected components (blocks) of  $G$  can be arranged in a path. By constructing counterexamples, we prove that no graph failing this condition (for example, a star) guarantees EF1, even for identical, additive, binary valuations. For the case of three or more agents, it is a challenging open problem to characterize the class of graphs guaranteeing EF1 (or even to find an infinite class of non-traceable graphs that guarantees EF1).

## 2 Preliminaries

For each natural number  $s \in \mathbb{N}$ , write  $[s] = \{1, 2, \dots, s\}$ .

Let  $N = [n]$  be a finite set of *agents* and  $G = (V, E)$  be an undirected finite graph. The vertices in  $V$  as *goods* or *items*. A subset  $I$  of  $V$  is *connected* if it induces a connected subgraph of  $G$ . We write  $\mathcal{C}(V)$  for the set of connected subsets of  $V$ . We call a set  $I \in \mathcal{C}(V)$  a (connected) *bundle*. Each agent  $i \in N$  has a *valuation function*  $u_i : \mathcal{C}(V) \rightarrow \mathbb{R}$  over connected bundles, which we will always assume to be *monotonic*, that is,  $X \subseteq Y$  implies  $u_i(X) \leq u_i(Y)$ . We also assume that  $u_i(\emptyset) = 0$  for each  $i \in N$ . Monotonicity implies that items are *goods*; we do not consider *bads* (or *chores*) in this paper. We say that an agent  $i \in N$  *weakly prefers* bundle  $X$  to bundle  $Y$  if  $u_i(X) \geq u_i(Y)$ .<sup>2</sup> A (connected) *allocation*  $A : N \rightarrow \mathcal{C}(V)$  assigns each agent  $i \in N$  a connected bundle  $A(i) \in \mathcal{C}(V)$  such that each item occurs in exactly one bundle, i.e.,  $\bigcup_{i \in N} A(i) = V$  and  $A(i) \cap A(j) = \emptyset$  when  $i \neq j$ .

We say that the agents have *identical valuations* if for all  $i, j \in N$  and every bundle  $I \in \mathcal{C}(V)$ , we have  $u_i(I) = u_j(I)$ . A valuation function  $u_i$  is *additive* if  $u_i(I) = \sum_{v \in I} u_i(\{v\})$  for each bundle  $I \in \mathcal{C}(V)$ . Many examples in this paper will use identical additive valuations, and will take  $G$  to be a path. In this case, we use a shorthand to specify these examples; the meaning of this notation should be clear. For example, we write “2–1–3–1” to denote an instance with four items  $v_1, v_2, v_3, v_4$  arranged on a path, and where  $u_i(\{v_1\}) = 2, \dots, u_i(\{v_4\}) = 1$  for each  $i$ . For such an instance, an allocation will be written as a tuple, e.g., (2, 1–3–1) denoting an allocation allocating bundles  $\{v_1\}$  and  $\{v_2, v_3, v_4\}$ , noting that with identical valuations it does not usually matter which agent receives which bundle.

<sup>2</sup> Our arguments only operate based on agents’ ordinal preferences over bundles, and the (cardinal) valuation functions are only used for notational convenience. One exception, perhaps, is in Algorithm 1 where we calculate a leximin allocation, but the algorithm can be applied after choosing an arbitrary utility function consistent with the ordinal preferences.



An allocation  $A$  is *envy-free* if  $u_i(A(i)) \geq u_i(A(j))$  for every pair  $i, j \in N$  of agents, that is, if every agent thinks that their bundle is a best bundle in the allocation. It is well-known that an envy-free allocation may not exist (consider two agents and one good). The main fairness notion that we study is a version of *envy-freeness up to one good* (EF1), a relaxation of envy-freeness introduced by Budish [12], adapted to the model with connectivity constraints. This property states that an agent  $i$  will not envy another agent  $j$  after we remove some item from  $j$ 's bundle. Since we only allow connected bundles in our set-up, we may only remove an item from  $A(j)$  if removal of this item leaves the bundle connected.

► **Definition 1** (EF1: envy-freeness up to one *outer* good). An allocation  $A$  satisfies *EF1* if for any pair  $i, j \in N$  of agents, either  $A(j) = \emptyset$  or there is a good  $v \in A(j)$  such that  $A(j) \setminus \{v\}$  is connected and  $u_i(A(i)) \geq u_i(A(j) \setminus \{v\})$ .

In the instance 2–1–3–1 for two agents, the allocation (2–1, 3–1) is EF1, since the left agent's envy can be eliminated by removing the item of value 3 from the right-hand bundle. However, the allocation (2, 1–3–1) fails to be EF1 according to our definition, since eliminating either outer good of the right bundle does not prevent envy.<sup>3</sup>

► **Definition 2.** A graph  $G$  *guarantees EF1* (for a specific  $n$ ) if for all possible monotonic valuations for  $n$  agents, there exists some connected allocation that is EF1.

For reasoning about EF1 allocations, let us introduce a few shorthands. Given an allocation  $A$  we will say that  $i \in N$  *does not envy*  $j \in N$  *up to*  $v$  if  $u_i(A(i)) \geq u_i(A(j) \setminus \{v\})$ . The *up-to-one valuation*  $u_i^- : \mathcal{C}(V) \rightarrow \mathbb{R}_{\geq 0}$  of agent  $i \in N$  is defined, for every  $I \in \mathcal{C}(V)$ , as

$$u_i^-(I) := \begin{cases} 0 & \text{if } I = \emptyset, \\ \min \{u_i(I \setminus \{v\}) : v \in I \text{ such that } I \setminus \{v\} \text{ is connected}\} & \text{if } I \neq \emptyset. \end{cases} \quad (1)$$

Thus, an allocation  $A$  satisfies EF1 iff  $u_i(A(i)) \geq u_i^-(A(j))$  for any pair  $i, j \in N$  of agents.

Given an ordered sequence of the vertices  $P = (v_1, v_2, \dots, v_m)$ , and  $j, k \in [m]$  with  $j \leq k$ , we write  $P(v_j, v_k)$  for the subsequence from  $v_j$  to  $v_k$ , so  $P(v_j, v_k) = (v_j, v_{j+1}, \dots, v_{k-1}, v_k)$ . Let  $L(v_j) = P(v_1, v_{j-1})$  be the subsequence of vertices strictly left of  $v_j$  and  $R(v_j) = P(v_{j+1}, v_m)$  be the subsequence of vertices strictly right of  $v_j$ . A *Hamiltonian path* of a graph  $G$  is a path that visits all the vertices of the graph exactly once. A graph is *traceable* if it contains a Hamiltonian path.

### 3 EF1 existence for two agents

In cake-cutting for two agents, the standard way of obtaining an envy-free allocation is the cut-and-choose protocol: Alice divides the cake into two equally-valued pieces, and Bob selects the piece he prefers; the other piece goes to Alice. The same strategy almost works in the indivisible case when items form a path; the problem is that Alice might not be able to divide the items into two exactly-equal pieces. Instead, we ask Alice to divide the items into pieces that are equally valued “up to one good”. The formal version is as follows. For a sequence of vertices  $P = (v_1, v_2, \dots, v_m)$  and an agent  $i$ , we say that  $v_j$  is the *lumpy tie* over  $P$  for agent  $i$  if  $j$  is the smallest index such that

$$u_i(L(v_j) \cup \{v_j\}) \geq u_i(R(v_j)) \quad \text{and} \quad u_i(R(v_j) \cup \{v_j\}) \geq u_i(L(v_j)). \quad (2)$$

<sup>3</sup> This example shows that our definition is strictly stronger than the standard definition of EF1 without connectivity constraints. In the instance 2–1–3–1, considered without connectivity constraints, the allocation (2, 1–3–1) does satisfy EF1 since in the standard setting we are allowed to remove the middle item (with value 3) of the right bundle.

For example, when  $i$  has additive valuations 1–3–2–1–3–1, then the third item (of value 2) is the lumpy tie for  $i$ , since  $1 + 3 + 2 \geq 1 + 3 + 1$  and  $2 + 1 + 3 + 1 \geq 1 + 3$ . The lumpy tie always exists: taking  $j$  to be the smallest index such that  $u_i(L(v_j) \cup \{v_j\}) \geq u_i(R(v_j))$  (which exists as the inequality holds for  $j = m$  by monotonicity), the first part of (2) holds. If  $j = 1$ , the second part of (2) is immediate by monotonicity. If  $j > 1$ , then since  $j$  is minimal, we have  $u_i(L(v_j)) = u_i(L(v_{j-1}) \cup \{v_{j-1}\}) < u_i(R(v_{j-1})) = u_i(R(v_j) \cup \{v_j\})$  as required.

Using lumpy ties, our discrete version of the cut-and-choose protocol is specified as follows.

**Discrete cut-and-choose protocol for  $n = 2$  agents** on a sequence

$P = (v_1, v_2, \dots, v_m)$ :

*Step 1.* Alice selects her lumpy tie  $v_j$  over  $(v_1, v_2, \dots, v_m)$ .

*Step 2.* Bob chooses a weakly preferred bundle among  $L(v_j)$  and  $R(v_j)$ .

*Step 3.* Alice receives the bundle of all the remaining vertices, including  $v_j$ .

Intuitively, the protocol allows Alice to select an item  $v_j$  that she will receive for sure, with the advice that the two pieces to either side of  $v_j$  should have almost equal value to her. Then, Bob is allowed to choose which side of  $v_j$  he wishes to receive. In our example with valuations 1–3–2–1–3–1, Alice selects the lumpy tie of value 2, then Bob chooses the bundle 1–3–1 to the right and receives it, and Alice receives the bundle 1–3–2. The result is EF1. This is true in general, and also if valuations are not identical.

► **Proposition 3.** *When  $G$  is a path and there are  $n = 2$  agents, the discrete cut-and-choose protocol yields an EF1 allocation.*

**Proof.** Clearly, the protocol returns a connected allocation. The returned allocation satisfies EF1: Bob does not envy Alice up to item  $v_j$ , since Bob receives his preferred bundle among  $L(v_j)$  and  $R(v_j)$ . Also, by (2), Alice does not envy Bob, since Alice either receives the bundle  $L(v_j) \cup \{v_j\}$  which she weakly prefers to Bob's bundle  $R(v_j)$ , or she receives the bundle  $R(v_j) \cup \{v_j\}$ , which she weakly prefers to Bob's bundle  $L(v_j)$ . ◀

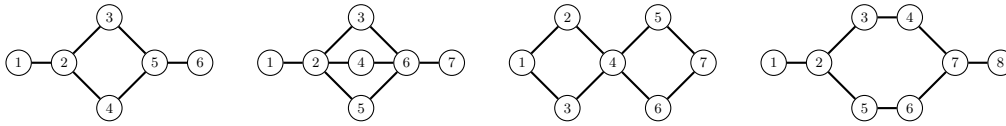
Proposition 3 implies that an EF1 allocation always exists on a path. Hence, an EF1 allocation exists for every traceable graph  $G$ : simply use the discrete cut-and-choose protocol on a Hamiltonian path of  $G$ . In fact, the discrete cut-and-choose protocol works on a broader class of graphs: We only need to require that the vertices of the graph can be numbered in a way that the allocation resulting from the discrete cut-and-choose protocol is guaranteed to be connected. Since the protocol always partitions the items into an initial and a terminal segment of the sequence, such a numbering needs to satisfy the following property.

► **Definition 4.** A *bipolar numbering* of a graph  $G$  is an ordering  $(v_1, v_2, \dots, v_m)$  of its vertices such that for all  $j \in [n]$ , the sets  $L(v_j) \cup \{v_j\}$  and  $R(v_j) \cup \{v_j\}$  are connected in  $G$ .

An equivalent definition (which is the standard one) says that a numbering is bipolar if for every  $j \in [n]$ , the vertex  $v_j$  has a neighbor that appears earlier in the sequence, and a neighbor that appears later in the sequence. Bipolar numberings are used in algorithms for testing planarity and for graph drawing. A Hamiltonian path induces a bipolar numbering, but there are non-traceable graphs that admit a bipolar numbering, see Figure 1 for examples.

► **Proposition 5.** *When there are  $n = 2$  agents, then the discrete cut-and-choose protocol run on a bipolar numbering of  $G$  yields an EF1 allocation.*

**Proof.** The discrete cut-and-choose protocol returns an allocation whose bundles are either initial or terminal segments of the ordered sequence  $(v_1, v_2, \dots, v_m)$ . By definition of a bipolar numbering, such an allocation is connected, and it is EF1 by Proposition 3. ◀

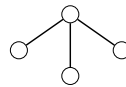


■ **Figure 1** Non-traceable graphs with bipolar numberings.

It is clear that the discrete cut-and-choose protocol cannot be extended to graphs other than those admitting a bipolar numbering. However, it could be that a different protocol is able to produce EF1 allocations on other graphs. In the remainder of this section, we prove that this is not the case: for  $n = 2$  agents, a connected graph  $G$  guarantees the existence of an EF1 allocation if and only if it admits a bipolar numbering. This completely characterizes the class of graphs that guarantee EF1 existence in the two-agent case.

### 3.1 Characterization of graphs guaranteeing EF1 for two agents

Based on a known characterization of graphs admitting a bipolar numbering, we characterize this class in terms of forbidden substructures. We then show that these forbidden structures are also forbidden for EF1: if a graph contains such a structure, we can exhibit an additive valuation profile for which no EF1 allocation exists.



As a simple example, consider the star with three leaves, which is the smallest connected graph that does not have a bipolar numbering. Take two agents with identical additive valuations that value each item at 1. Any connected allocation must allocate three items to one agent, and a single item to the other agent. Then the latter agent envies the former agent, even up to one good. This star is an example of what we call a *trident*, and forms a forbidden substructure. Informally, the forbidden substructures take one of two forms. We will prove that a graph  $G$  fails to admit a bipolar numbering, and fails to guarantee EF1 for two agents, iff either

- (a) there is a vertex  $s$  whose removal from  $G$  leaves three or more connected components, or
- (b) there are subgraphs  $C, P_1, P_2, P_3$  of  $G$  such that (i)  $G$  is the union of these subgraphs, (ii) the subgraphs  $P_1, P_2, P_3$  are vertex-disjoint, (iii)  $C$  has exactly one vertex in common with  $P_i$ ,  $i = 1, 2, 3$ , and (iv) no edge joins a vertex from one subgraph to a different one.

To reason about these structures, it is useful to consider the *block decomposition* of a graph, which will show that graphs that admit a bipolar numbering have a path-like structure.

► **Definition 6.** A vertex is called a *cut* vertex of a graph  $G$  if removing it increases the number of connected components of  $G$ . A graph  $G$  is *biconnected* if  $G$  does not have a cut vertex. A *block* of  $G$  is a maximal biconnected subgraph of  $G$ .

Equivalently, a block of a graph  $G$  can be defined as a maximal subgraph of  $G$  where each pair of vertices lie on a common cycle [8]. Given a connected graph  $G$ , we define a bipartite graph  $B(G)$  with bipartition  $(\mathcal{B}, \mathcal{S})$ , where  $\mathcal{B}$  is the set of blocks of  $G$  and  $\mathcal{S}$  is the set of cut vertices of a graph  $G$ ; a block  $B$  and a cut vertex  $v$  are adjacent in  $B(G)$  if and only if  $B$  includes  $v$ . Since every cycle of a graph is included in some block, the graph  $B(G)$  is a tree:

- **Lemma 7** (e.g., [8]). *Let  $G$  be a connected graph. Then*
- *any two blocks of  $G$  have at most one cut vertex in common;*
  - *the set of blocks forms a decomposition of  $G$ ; and*
  - *the graph  $B(G)$  is a tree.*

Thus, for a connected graph  $G$ , we call  $B(G)$  the *block tree* of  $G$ . It turns out that  $G$  admits a bipolar numbering if and only if  $B(G)$  is a path. For example, the graphs shown in Figure 1 all have their blocks arranged in a path (so that  $B(G)$  is a path).

- **Lemma 8.** *A graph  $G$  admits a bipolar numbering if its block tree  $B(G)$  is a path.*

**Proof.** Lempel et al. [20] show that  $G$  admits a bipolar numbering if there are  $s, t \in V$  such that adding an edge  $\{s, t\}$  to  $G$  makes it biconnected. If  $B(G)$  is a path, let  $B_1$  and  $B_2$  be the leaf blocks at the ends of the path  $B(G)$ . Take any  $s \in B_1$  and  $t \in B_2$ . If we add the edge  $\{s, t\}$  to  $G$ , the graph becomes biconnected. Hence,  $G$  admits a bipolar numbering. ◀

There is a linear-time algorithm based on depth-first search to construct a bipolar numbering for any biconnected graph [16, 31], and one can also calculate the block tree  $B(G)$  of a given graph in linear time [17]. Thus, in linear time, we can compute a bipolar numbering of a graph or report that none exists. Clearly, given a bipolar numbering, the discrete cut-and-choose protocol can also be run in linear time.

Next, we show that if  $B(G)$  is not a path, then  $G$  cannot guarantee EF1. The proof constructs explicit counter-examples, which have a very simple structure. We say that additive valuations  $u_i$  are *binary* if  $u_i(\{v\}) \in \{0, 1\}$  for every  $v \in V$ .

- **Lemma 9.** *Let  $G$  be a connected graph whose block tree  $B(G)$  is not a path. Then there exist identical, additive, binary valuations over  $G$  for two agents such that no connected allocation is EF1.*

**Proof.** If  $B(G)$  is not a path, then  $B(G)$  has a *trident*, i.e., a vertex with at least three neighbors, and thus either

- (a) there is a cut vertex  $s$  adjacent to three blocks  $B_1$ ,  $B_2$ , and  $B_3$ ; or
- (b) there is a block  $B$  adjacent to three different cut vertices  $s_1$ ,  $s_2$ , and  $s_3$ .

In either case, we construct identical additive valuations that do not admit an EF1 allocation.

In case (a), for each  $i = 1, 2, 3$ , choose a vertex  $v_i$  from  $B_i \setminus \{s\}$ . Note that we can choose such  $v_i \neq s$  due to the maximality of  $B_i$ . The two agents have utility 1 for  $s$ ,  $v_1$ ,  $v_2$ , and  $v_3$ , and 0 for the remaining vertices. Now take any connected allocation  $(I_1, I_2)$ . One of the bundles, say  $I_1$ , includes the cut vertex  $s$ . Then  $I_2$  can contain at most one of the vertices  $v_1$ ,  $v_2$ ,  $v_3$ , since  $I_2$  is connected and does not contain  $s$  yet any path between distinct  $v_i$  and  $v_j$  goes through  $s$ . Hence  $u_i(I_2) \leq 1$ . Now, the bundle  $I_1$  contains  $s$  and at least two of  $v_1$ ,  $v_2$ ,  $v_3$ , so  $u_i(I_1) \geq 3$ . Thus, the allocation is not EF1.

Case (b) is handled similarly; see [5]. ◀

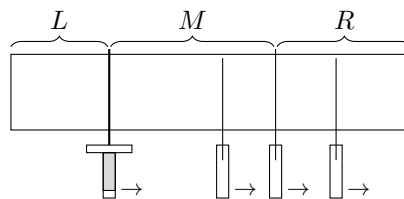
Combining these results, we obtain the promised characterization.

- **Theorem 10.** *The following conditions are equivalent for every connected graph  $G$ :*

1.  $G$  admits a bipolar numbering.
2.  $G$  guarantees EF1 for two agents.
3.  $G$  guarantees EF1 for two agents with identical, additive, binary valuations.
4. The block tree  $B(G)$  is a path.

The equivalence (2)  $\Leftrightarrow$  (3) is noteworthy and perhaps surprising: It is often easier to guarantee fairness when agents' valuations are identical, yet in terms of the graphs that guarantee EF1 for two agents, there is no difference between identical and non-identical valuations. Intriguingly, even for more than two agents, we do not know of a graph which guarantees EF1 for identical valuations, but fails it for non-identical valuations.

#### 4 EF1 existence for three agents: A moving-knife protocol



We will now consider the case of three agents. Stromquist [28] designed a protocol that results in an envy-free contiguous allocation of a divisible cake. In outline, the protocol works as follows. A referee holds a sword over the cake. Each of the three agents holds their own knife over the portion of the cake to the right of the sword, positioning it so that this portion is divided into two pieces they judge to have the same value. Now, initially, the sword is at the left end of the cake. It starts moving at constant speed from left to right, while the agents continuously move their knives to keep dividing the right-hand portion into equally-valued pieces. At some point (when the left-most piece becomes valuable enough), one of the agents shouts “cut”, and the cake will be cut twice: once by the sword, and once by the middle one of the three knives. Agents shout “cut” as soon as the left piece is a highest-valued piece among the three. The agent who shouts receives the left piece. The remaining agents each receive a piece containing their knife. The resulting allocation is envy-free, since the agent receiving the left piece prefers it to the other pieces, and the other agents who are not shouting receive at least half the value of the part of the cake to the right of the sword.

Let  $G$  be a path,  $P = (v_1, v_2, \dots, v_m)$ . There are several difficulties in translating Stromquist’s continuous procedure to the discrete setting for  $G$ . First, agents need to divide the piece to the right of the sword in half, and this might not be possible exactly given indivisibilities; but this can be handled using our concept of lumpy ties from Section 3. Next, when the sword moves one item to the right, the lumpy ties of the agents may need to jump several items to the right, for example because the new member of the left-most bundle is very valuable. To ensure EF1, we will need to smoothen these jumps, so that the middle piece grows one item at a time. Also, it will be helpful to have the sword move in half-steps: it alternates between being placed between items (so it cuts the edge between the items), and being placed over an item, in which case the sword covers the item and agents ignore that item. Finally, while the sword covers an item, we will only terminate if at least *two* agents shout to indicate that they prefer the left-most piece; this will ensure that there is an agent who is flexible about which of the bundles they are assigned. The algorithm moves in steps, and alternates between moving the sword, and updating the lumpy ties.

In our formal description of the algorithm, we do not use swords and knives. Instead, we maintain three bundles  $L$ ,  $M$ , and  $R$  that can be seen as resulting from a certain configuration of these cutting implements. We also need a few definitions. For a subsequence of vertices  $P(v_s, v_r) = (v_s, v_{s+1}, \dots, v_r)$  and an agent  $i$ , recall that  $v_j$  ( $s \leq j \leq r$ ) is the *lumpy tie* over

$P(v_s, v_r)$  for  $i$  if  $j$  is the smallest index such that

$$u_i(L(v_j) \cup \{v_j\}) \geq u_i(R(v_j)) \quad \text{and} \quad u_i(R(v_j) \cup \{v_j\}) \geq u_i(L(v_j)). \quad (3)$$

Here, the definitions of  $L(v_j)$  and  $R(v_j)$  apply to the subsequence  $P(v_s, v_r)$ . The lumpy tie always exists by the discussion after equation (2). Each of the three agents has a lumpy tie over  $P(v_s, v_r)$ ; a key concept for us is the *median lumpy tie* which is the median of the lumpy ties of the three agents, where the median is taken with respect to the ordering of  $P(v_s, v_r)$ . We say that  $i \in N$  is a *left agent* (respectively, a *middle agent* or a *right agent*) over  $P(v_s, v_r)$  if the lumpy tie for  $i$  appears strictly before (respectively, is equal to, or appears strictly after) the median lumpy tie. Note that by definition of median, there is at most one left agent, at most one right agent, and at least one middle agent.

Suppose that the median lumpy tie over the subsequence  $P(v_s, v_r)$  is  $v_j$ , and let  $i$  be an agent. Then using the definitions of lumpy tie and left/right agents, we find that

$$\begin{aligned} u_i(L(v_j)) &\geq u_i(R(v_j) \cup \{v_j\}) && \text{if } i \text{ is a left agent, and} \\ u_i(R(v_j)) &\geq u_i(L(v_j) \cup \{v_j\}) && \text{if } i \text{ is a right agent.} \end{aligned} \quad (4)$$

Given the median lumpy tie  $v_j$  over  $P(v_s, v_r)$ , and a two-agent set  $S = \{i, k\} \subseteq N$ , we define  $\text{Lumpy}(S, v_j, P(v_s, v_r))$  to be the allocation of the items in  $P(v_s, v_r)$  to  $S$  such that

- if  $i$  is a left agent and  $k$  is a right agent, then  $i$  receives  $L(v_j)$  and  $k$  receives  $R(v_j) \cup \{v_j\}$ ;
- if  $i$  is a middle agent, then agent  $k$  receives  $k$ 's preferred bundle among  $L(v_j)$  and  $R(v_j)$ , and agent  $i$  receives the other bundle along with  $v_j$ .

Using (3) and (4), we see that  $\text{Lumpy}(S, v_j, P(v_s, v_r))$  is an EF1 allocation:

► **Lemma 11** (Median Lumpy Ties Lemma). *Let  $S = \{i, k\} \subseteq N$  and let  $v_j$  be the median lumpy tie over  $P(v_s, v_r)$ . Then  $\text{Lumpy}(S, v_j, P(v_s, v_r))$  is an EF1 allocation of the items in  $P(v_s, v_r)$  to  $S$ . Further, each agent in  $S$  weakly prefers their bundle to  $L(v_j)$  and  $R(v_j)$ .*

The algorithm is specified in Figure 2. It alternately moves a left pointer  $\ell$  (in Steps 2 and 3) and a right pointer  $r$  (in Step 4). See [5] for a proof of the following theorem.

► **Theorem 12.** *The moving-knife protocol in Figure 2 finds an EF1 allocation for three agents and runs in  $O(|V|)$  time, when  $G$  is traceable.*

## 5 EF2 existence for any number of agents

For two or three agents, we have seen algorithms that are guaranteed to find an EF1 allocation on a path (and on traceable graphs). Both algorithms were adaptations of procedures that identify envy-free divisions in the cake-cutting problem. For the case of four or more agents, we face a problem: there are no known procedures that find connected envy-free division in cake-cutting if the number of agents is larger than three. However, in the divisible setting, a non-constructive existence result is known: Su [29] proved, using Sperner's lemma, that for any number of agents, a connected envy-free division of a cake always exists. One might try to use this result as a black box to obtain a fair allocation for the indivisible problem on a path: Translate an indivisible instance with additive valuations into a divisible cake (where each item corresponds to a region of the cake), obtain an envy-free division of the cake, and round it to get an allocation of the items. Suksompong [30] followed this approach and showed that the result is an allocation where any agent  $i$ 's envy  $u_i(A(j)) - u_i(A(i))$  is at most  $2u_{\max}$ , where  $u_{\max}$  is the maximum valuation for a single item.

In this section, rather than using Su's [29] result as a black box, we directly apply Sperner's lemma to the indivisible problem. This allows us to obtain a stronger fairness guarantee: We show that on paths (and on traceable graphs), there always exists an EF2

**Discrete moving-knife protocol for  $n = 3$  agents** on a sequence  $P = (v_1, v_2, \dots, v_m)$ :  
 An agent  $i \in N$  is a *shouter* if  $u_i(L) \geq u_i(M)$  and  $u_i(L) \geq u_i(R)$ .

- Step 1.* Initialize  $\ell = 0$  and set  $r$  so that  $v_r$  is the median lumpy tie over the subsequence  $P(v_2, v_m)$ . Initialize  $L = \emptyset$ ,  $M = \{v_2, v_3, \dots, v_{r-1}\}$ , and  $R = \{v_{r+1}, v_{r+2}, \dots, v_m\}$ .
- Step 2.* Add an additional item to  $L$ , i.e., set  $\ell = \ell + 1$  and  $L = \{v_1, v_2, \dots, v_\ell\}$ . If no agent shouts, go to Step 3. If some agent  $s_{\text{left}}$  shouts,  $s_{\text{left}}$  receives the left bundle  $L$ . Allocate the remaining items according to  $\text{Lumpy}(N \setminus \{s_{\text{left}}\}, v_r, P(v_{\ell+1}, v_m))$ .
- Step 3.* Delete the left-most point of the middle bundle, i.e., set  $M = \{v_{\ell+2}, v_{\ell+3}, \dots, v_{r-1}\}$ . If the number of shouters is smaller than two, go to Step 4. If at least two agents shout, we show (next page) that there is a shouter  $s$  who is a middle agent over  $P(v_{\ell+1}, v_m)$ . Then, allocate  $L$  to a shouter  $s_{\text{left}}$  distinct from  $s$ . Let the agent  $c$  distinct from  $s$  and  $s_{\text{left}}$  choose his preferred bundle among  $\{v_{\ell+1}\} \cup M$  and  $\{v_r\} \cup R$ . Agent  $s$  receives the other bundle.
- Step 4.* If  $v_r$  is the median lumpy tie over  $P(v_{\ell+2}, v_m)$ , directly move to the following cases (a)–(d). If  $v_r$  is not the median lumpy tie over  $P(v_{\ell+2}, v_m)$ , set  $r = r + 1$ ,  $M = \{v_{\ell+2}, v_{\ell+3}, \dots, v_{r-1}\}$ , and  $R = \{v_{r+1}, v_{r+2}, \dots, v_m\}$ ; then, go to cases (a)–(d).
- a. If at least two agents shout, find a shouter  $s$  who did not shout at the previous step. If there is a shouter  $s_{\text{left}}$  who shouted at the previous step,  $s_{\text{left}}$  receives  $L$ ; else, give  $L$  to an arbitrary shouter  $s_{\text{left}}$  distinct from  $s$ . The agent  $c$  distinct from  $s$  and  $s_{\text{left}}$  choose his preferred bundle among  $\{v_{\ell+1}\} \cup M$  and  $\{v_r\} \cup R$ , breaking ties in favor of the former option. Agent  $s$  receives the other bundle.
  - b. If  $v_r$  is the median lumpy tie over  $P(v_{\ell+2}, v_m)$  and only one agent  $s_{\text{left}}$  shouts, give  $L \cup \{v_{\ell+1}\}$  to  $s_{\text{left}}$  and allocate the rest according to  $\text{Lumpy}(N \setminus \{s_{\text{left}}\}, v_r, P(v_{\ell+2}, v_m))$ .
  - c. If  $v_r$  is the median lumpy tie over  $P(v_{\ell+2}, v_m)$  but no agent shouts, go to Step 2.
  - d. Otherwise  $v_r$  is not the median lumpy tie over  $P(v_{\ell+2}, v_m)$ : Repeat Step 4.

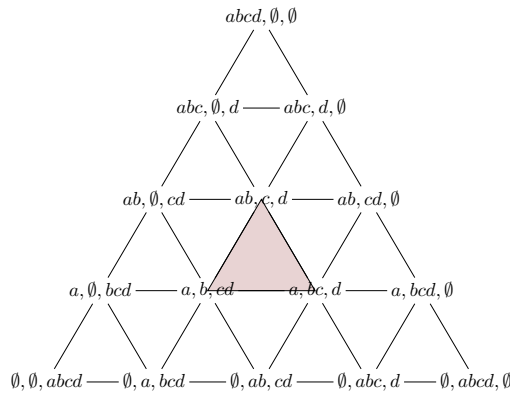
■ **Figure 2** Algorithm producing EF1 allocations for three agents.

allocation.<sup>4</sup> An allocation is EF2 if any agent's envy can be avoided by removing up to two items from the envied bundle. Again, we only allow removal of items if this operation leaves a connected bundle. For example, on a path, if agent  $i$  envies the bundle of agent  $j$ , then  $i$  does not envy that bundle once we remove its two endpoints.

► **Definition 13** (EF2: envy-freeness up to two outer goods). An allocation  $A$  satisfies EF2 if for any pair  $i, j \in N$  of agents, either  $|A(j)| \leq 1$ , or there are two goods  $u, v \in A(j)$  such that  $A(j) \setminus \{u, v\}$  is connected and  $u_i(A(i)) \geq u_i(A(j) \setminus \{u, v\})$ .

<sup>4</sup> To see that EF2 is a stronger property than bounding envy up to  $2u_{\max}$ , consider a path of four items and two agents with additive valuations 1–10–2–2. The allocation (1, 10–2–2) is not EF2, but the first agent has an envy of  $13 < 20 = 2u_{\max}$ .





Let us first give a high-level illustration with three agents of how Sperner’s lemma can be used to find low-envy allocations. Given a path, say  $P = (a, b, c, d)$ , the family of connected partitions of  $P$  can naturally be arranged as the vertices of a subdivided simplex, as in the figure on the right. For each of these partitions, each agent  $i$  labels the corresponding vertex by the index of a bundle from that partition that  $i$  most-prefers. For example, the top vertex will be labelled as “index 1” by all agents, since they all most-prefer the left-most bundle in  $(abcd, \emptyset, \emptyset)$ . Now, Sperner’s lemma will imply that at least one of the simplices (say the shaded one) is “fully-labeled”, which means that the first agent most-prefers the left-most bundle at one vertex, the second agent most-prefers the middle bundle at another vertex, and the third agent most-prefers the right-most bundle at the last vertex. Notice that the partitions at the corner points of the shaded simplex are all “similar” to each other (they can be obtained from each other by moving only 1 item). Hence, we can “round” the corner-partitions into a common allocation  $A^*$ , say by picking one of the corner partitions arbitrarily and then allocating bundles to agents according to the labels. The resulting allocation has the property that any agents’ envy can be eliminated by moving at most one good.<sup>5</sup>

The argument sketched above does not yield an EF1 nor even an EF2 allocation. Intuitively, the problem is that the connected partitions at the corners of the fully-labeled simplex are “too far apart”, so that no matter how we round the corner partitions into a common allocation  $A^*$ , some agents’ bundles will have changed too much, and so we cannot prevent envy even up to one or two goods. In the following, we present a solution to this problem, by considering a finer subdivision: we introduce  $n - 1$  knives which move in half-steps (rather than full steps), and which might “cover” an item so that it appears in none of the bundles. The result is that the partial partitions in the corners of the fully-labeled simplex are closer together, and can be successfully rounded into an EF2 allocation  $A^*$ .

### 5.1 Sperner’s lemma

We start by formally introducing Sperner’s lemma. Let  $\text{conv}(\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_k)$  denote the convex hull of  $k$  vectors  $\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_k$ . An  $n$ -simplex is an  $n$ -dimensional polytope which is the convex hull of its  $n + 1$  main vertices. A  $k$ -face of the  $n$ -simplex is the  $k$ -simplex formed by the span of any subset of  $k + 1$  main vertices. A triangulation  $T$  of a simplex  $S$  is a

<sup>5</sup> One can generalize this argument to show that on paths, there exists an allocation  $A$  satisfying a weak form of EF1: for any  $i, j \in [n]$ , we have  $u_i(I_i \cup \{g_i\}) \geq u_i(I_j \setminus \{g_j\})$  for some items  $g_i, g_j$  such that  $I_i \cup \{g_i\}$  and  $I_j \setminus \{g_j\}$  are connected. For additive valuations, this implies that envy is bounded by  $u_i(g_i) + u_i(g_j) \leq 2u_{\max}$ , which is Suksompong’s [30] result.



collection of sub- $n$ -simplices whose union is  $S$  with the property that the intersection of any two of them is either the empty set, or a face common to both. Each of the sub-simplices  $S^* \in T$  is called an *elementary* simplex of the triangulation  $T$ . We denote by  $V(T)$  the set of vertices of the triangulation  $T$ , i.e., the union of vertices of the elementary simplices of  $T$ .

Let  $T$  be some fixed triangulation of an  $(n - 1)$ -simplex  $S = \text{conv}(\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_n)$ . A *labeling function* is a function  $L : V(T) \rightarrow [n]$  that assigns a number in  $[n]$  (called a *color*) to each vertex of the triangulation  $T$ . A labeling function  $L$  is called *proper* if

- For each main vertex  $\mathbf{v}_i$  of the simplex,  $L$  assigns color  $i$  to  $\mathbf{v}_i$ :  $L(\mathbf{v}_i) = i$ ; and
  - $L(\mathbf{v}) \neq i$  for any vertex  $\mathbf{v} \in V(T)$  belonging to the  $(n - 2)$ -face of  $S$  not containing  $\mathbf{v}_i$ .
- Sperner's lemma states that if  $L$  is a proper labeling function, then there exists an elementary simplex of  $T$  whose vertices all have different labels.

We will consider a generalized version of Sperner's lemma, proved, for example, in [2]. In this version, there are  $n$  labeling functions  $L_1, \dots, L_n$ , and we are looking for an elementary simplex which is *fully-labeled* for some way of assigning labeling functions to vertices, where we must use each labeling function exactly once. The formal definition is as follows.

► **Definition 14** (Fully-labeled simplex). Let  $T$  be a triangulation of an  $(n - 1)$ -simplex, and let  $L_1, \dots, L_n$  be labeling functions. An elementary simplex  $S^*$  of  $T$  is *fully-labeled* if we can write  $S^* = \text{conv}(\mathbf{v}_1^*, \mathbf{v}_2^*, \dots, \mathbf{v}_n^*)$  such that there exists a permutation  $\phi : [n] \rightarrow [n]$  with

$$L_i(\mathbf{v}_i^*) = \phi(i) \quad \text{for each } i \in [n].$$

► **Lemma 15** (Generalized Sperner's Lemma). Let  $T$  be a triangulation of an  $(n - 1)$ -simplex  $S$ , and let  $L_1, \dots, L_n$  be proper labeling functions. Then there is a fully-labeled simplex  $S^*$  of  $T$ .

## 5.2 Existence of EF2 allocations

Suppose that our graph  $G$  is a path  $P = (1, 2, \dots, m)$ , where the items are named by integers. We assume that  $m \geq n$ , so that there are at least as many items as agents (when  $m < n$  it is easy to find EF1 allocations). Our aim is to cut the path  $P$  into  $n$  intervals (bundles)  $I_*^1, I_*^2, \dots, I_*^n$ . Throughout the argument, we use superscripts to denote indices of bundles; index 1 refers to the left-most bundle and index  $n$  refers to the right-most bundle.

**Construction of the triangulation.** Consider the  $(n - 1)$ -simplex

$$S_m = \{ \mathbf{x} \in \mathbb{R}^{n-1} : \frac{1}{2} \leq x^1 \leq x^2 \leq \dots \leq x^{n-1} \leq m + \frac{1}{2} \}. \quad (5)$$

We construct a triangulation  $T_{\text{half}}$  of  $S_m$  whose vertices  $V(T_{\text{half}})$  are the points  $\mathbf{x} \in S_m$  such that each  $x^j$  is either integral or half-integral, namely,

$$V(T_{\text{half}}) = \{ \mathbf{x} \in S_m : x^j \in \{ \frac{1}{2}, 1, \frac{3}{2}, 2, \frac{5}{2}, \dots, m, m + \frac{1}{2} \} \text{ for all } j \in [n] \}.$$

For reasons that will become clear shortly, we call a vector  $\mathbf{x} \in V(T_{\text{half}})$  a *knife position*.

Using Kuhn's triangulation [19, 26, 14], we construct  $T_{\text{half}}$  so we can write each elementary simplex  $S' \in T_{\text{half}}$  as  $S' = \text{conv}(\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n)$  and there is a permutation  $\pi : [n] \rightarrow [n]$  with

$$\mathbf{x}_{i+1} = \mathbf{x}_i + \frac{1}{2} \mathbf{e}^{\pi(i)} \quad \text{for each } i \in [n - 1], \quad (6)$$

where  $\mathbf{e}^j = (0, \dots, 1, \dots, 0)$  is the  $j$ -th unit vector. We give an interpretation of (6) shortly.

Each vertex  $\mathbf{x} = (x^1, x^2, \dots, x^{n-1}) \in V(T_{\text{half}})$  of the triangulation  $T_{\text{half}}$  corresponds to a partial partition  $A(\mathbf{x}) = (I^1(\mathbf{x}), I^2(\mathbf{x}), \dots, I^n(\mathbf{x}))$  of  $P$  where  $I^j(\mathbf{x}) := \{y \in \{1, 2, \dots, m\} : x^{j-1} < y < x^j\}$ , writing  $x^0 = \frac{1}{2}$  and  $x^n = m + \frac{1}{2}$  for convenience. Intuitively,  $\mathbf{x}$  specifies the

location of  $n - 1$  knives that cut  $P$  into  $n$  pieces. If  $x^j$  is integral, that is  $x^j \in \{1, \dots, m\}$ , then the  $j$ -th knife “covers” the item  $x^j$ , which is then part of neither  $I^j(\mathbf{x})$  nor  $I^{j+1}(\mathbf{x})$ . This is why  $A(\mathbf{x})$  is a *partial* partition. Since there are only  $n - 1$  knives but  $m \geq n$  items, not all items are covered, so at least one bundle is non-empty.

Property (6) means that, if we visit the knife positions  $\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n$  at the corners of an elementary simplex in the listed order, then at each step exactly one of the knives moves by half a step, and each knife moves only at one of the steps.

**Construction of the labeling functions.** We now construct, for each agent  $i \in [n]$ , a labeling function  $L_i : V(T_{\text{half}}) \rightarrow [n]$ . The function  $L_i$  takes as input a vertex  $\mathbf{x}$  of the triangulation  $T_{\text{half}}$  (interpreted as the partial partition  $A(\mathbf{x})$ ), and returns a color in  $[n]$ . The color will specify the index of a bundle in  $A(\mathbf{x})$  that agent  $i$  likes most. Formally,

$$L_i(\mathbf{x}) \in \{j \in [n] : u_i(I^j(\mathbf{x})) \geq u_i(I^k(\mathbf{x})) \text{ for all } k \in [n]\}.$$

If there are several most-preferred bundles in  $A(\mathbf{x})$ , ties can be broken arbitrarily. However, we insist that the index  $L_i(\mathbf{x})$  always corresponds to a non-empty bundle (this can be ensured since  $A(\mathbf{x})$  always contains a non-empty bundle, and  $u_i$  is monotonic).

The labeling functions  $L_i$  are proper. For each  $j \in [m]$ , the main vertex  $\mathbf{v}_j$  of the simplex  $S_m$  has the form  $\mathbf{v}_j = (\frac{1}{2}, \dots, \frac{1}{2}, m + \frac{1}{2}, \dots, m + \frac{1}{2})$ , where the first  $j - 1$  entries are  $\frac{1}{2}$  and the rest are  $m + \frac{1}{2}$ . In the partition  $A(\mathbf{v}_j)$ , the bundle  $I^j(\mathbf{v}_j)$  contains all the items, so is most-preferred (since  $u_i$  is monotonic and by our tie-breaking), and so  $L_i(\mathbf{v}_j) = j$ . Further, any vertex  $\mathbf{x}$  belonging to the  $(n - 2)$ -face of  $S_m$  not containing  $\mathbf{v}_j$  satisfies  $x^{j-1} = x^j$ , and thus in partition  $A(\mathbf{x})$ , bundle  $I^j(\mathbf{x})$  is empty, hence is *not* selected, and so  $L_i(\mathbf{x}) \neq j$ .

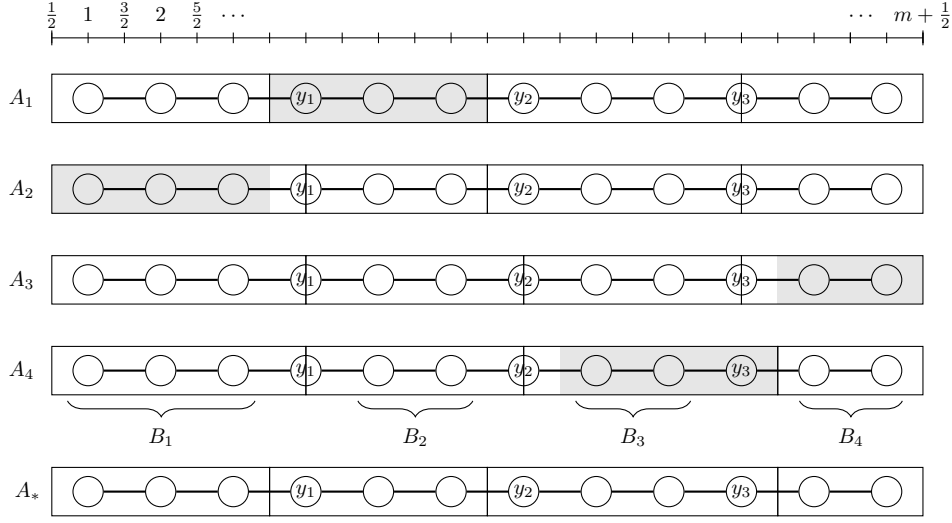
By the generalized version of Sperner’s lemma (Lemma 15), there exists an elementary simplex  $S^* = \text{conv}(\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n)$  of the triangulation  $T_{\text{half}}$  which is fully-labeled, so that, for some permutation  $\phi : [n] \rightarrow [n]$ , we have  $L_i(\mathbf{x}_i) = \phi(i)$  for all  $i \in [n]$ .

**Translation into partial partitions.** The fully-labeled elementary simplex  $S^*$  corresponds to a sequence  $(A_1, A_2, \dots, A_n)$  of partial partitions of  $P$ , which we call the *Sperner sequence*, where  $A_i = (I_i^1, \dots, I_i^n) := A(\mathbf{x}_i)$  for each  $i \in [n]$ . An example of a Sperner sequence is shown in Figure 3. From the labeling, for each agent  $i \in [n]$ , since  $L_i(\mathbf{x}_i) = \phi(i)$ , the bundle with index  $\phi(i)$  in the partition  $A_i$  is a best bundle for  $i$ :

$$u_i(I_i^{\phi(i)}) \geq u_i(I_i^j) \quad \text{for each } j \in [n]. \quad (7)$$

Now, for each  $j \in [n]$ , we define the *basic bundle*  $B^j := I_1^j \cap \dots \cap I_n^j$  to be the bundle of items that appear in the  $j$ -th bundle of every partition in the Sperner sequence. The set of basic bundles is a partial partition. Let us analyze the items between basic bundles. From (6), each of the  $n - 1$  knives moves exactly once, by half a step, while passing through the Sperner sequence  $(A_1, A_2, \dots, A_n)$ . Thus, the numbers  $x_1^j, \dots, x_n^j$  take on two different values, one of which is integral and the other half-integral. We write  $y^j$  for the integral value (so  $y^j = x_i^j$  for some  $i \in [n]$ ), and call  $y^j$  a *boundary item*. The  $j$ -th knife covers the item  $y^j$  in some, but not all, of the partial partitions in the Sperner sequence. Now, there are two cases:

- (a)  $x_1^j = \dots = x_i^j = y^j - \frac{1}{2}$  and  $x_{i+1}^j = \dots = x_n^j = y^j$  for some  $i \in [n]$ , so that  $y^j$  never occurs in the  $j$ -th bundle in the Sperner sequence but sometimes occurs in the  $j + 1$ st bundle, or



■ **Figure 3** Example of the Sperner sequence  $A_1, \dots, A_4$  for  $n = 4$ , as well as the derived partition  $A_*$ . Vertical lines indicate the positions  $x_i^1, x_i^2, x_i^3$  of the knives,  $i = 1, \dots, 4$ . Shaded in gray, for  $i = 1, \dots, 4$ , is the bundle  $I_i^{\phi(i)}$  selected by agent  $i$  as their favorite bundle in  $A_i$ .

(b)  $x_1^j = \dots = x_i^j = y^j$  and  $x_{i+1}^j = \dots = x_n^j = y^j + \frac{1}{2}$  for some  $i \in [n]$ , so that  $y^j$  sometimes occurs in the  $j$ -th bundle in the Sperner sequence but never occurs in the  $j + 1$ st bundle. Since  $y^j$  is sometimes covered by a knife, it is not part of any basic bundle. Note that

$$B^j \subseteq I_i^j \subseteq \{y^{j-1}\} \cup B^j \cup \{y^j\} \quad \text{for every } i, j \in [n]. \quad (8)$$

**Rounding into a complete partition.** We now construct a complete partition of the path  $P$  into the bundles  $(I_*^1, I_*^2, \dots, I_*^n)$  which are defined as follows:

$$I_*^j := I_1^j \cup \dots \cup I_n^j \quad \text{for each } j \in [n].$$

Thus, the bundle  $I_*^j$  contains the basic bundle  $B^j$ , plus all of the boundary items  $y^{j-1}$  or  $y^j$  that occur in the  $j$ -th bundle at some point of the Sperner sequence. Precisely, for each boundary item  $y^j$ ,  $j \in [n - 1]$ , the item  $y^j$  is placed in bundle  $I_*^{j+1}$  in case (a) above, and it is placed in bundle  $I_*^j$  in case (b). Thus, every item is allocated to exactly one bundle.

**An EF2 allocation.** We first show that the partition  $(I_*^1, I_*^2, \dots, I_*^n)$  is such that agents' expectations about the value of the bundles  $I_*^j$  are approximately correct (up to two items):

$$u_i(I_*^j) \geq u_i(I_i^j) \geq u_i(B^j) \quad \text{for every agent } i \in [n] \text{ and every } j \in [n]. \quad (9)$$

This follows by monotonicity of  $u_i$ , since  $I_*^j = I_1^j \cup \dots \cup I_n^j \supseteq I_i^j \supseteq B^j$  by (8).

Now, based on the partition, we define an allocation  $A_*$  by  $A_*(i) = I_*^{\phi(i)}$  for each agent  $i \in [n]$ . Then  $A_*$  satisfies EF2: For any pair  $i, j \in [n]$  of agents, we have

$$u_i(A_*(i)) = u_i(I_*^{\phi(i)}) \stackrel{(9)}{\geq} u_i(I_i^{\phi(i)}) \stackrel{(7)}{\geq} u_i(I_i^{\phi(j)}) \stackrel{(9)}{\geq} u_i(B^{\phi(j)}) \stackrel{(8)}{=} u_i(A_*(j) \setminus \{y^{j-1}, y^j\}).$$

Hence, we have proved the main result of this section:

► **Theorem 16.** *On a path, for any number of agents with monotone valuation functions, a connected EF2 allocation exists.*

## 6 EF1 existence for four agents

We have seen that Sperner's lemma can be used to show EF2 existence for any number of agents. Why does our proof in the previous section only establish EF2, and not EF1? The reason is that agents' expectations about the contents of a bundle might differ by up to *two* goods from what the bundle will actually contain. In the notation of the previous section, an agent  $i$  may be presented with a partial partition  $I_i$  where the  $j$ -th bundle  $I_i^j$  is the basic bundle, i.e.,  $I_i^j = B^j$ . The agent then selects their favorite bundle from  $I_i$ , implicitly assuming that the  $j$ -th bundle in the rounded partition  $I_*$  will also equal  $B^j$ , i.e., that  $I_*^j = B^j$ . However, it may happen that in fact  $I_*^j = \{y^{j-1}\} \cup B^j \cup \{y^j\}$ , and then  $i$  envies the agent who receives bundle  $j$  by a margin of two goods.

For four agents, we can adapt our argument to achieve EF1. To do this, we both change the way we round the Sperner sequence into an allocation, and we define new labeling functions that better anticipate how a partial partition will be rounded into the final allocation. In this way, agents' expectations about bundles will only ever be wrong up to one good. In crude terms, agents will expect that each of the two interior bundles will be assigned at least one of the boundary items, and the rounding method ensures that this will indeed happen.

Let  $n = 4$ . Formally, to define the labeling function, for each agent  $i \in [n]$  we construct a *virtual valuation function*  $\hat{u}_i(\mathbf{x}, j)$  which assigns a value to each bundle  $j \in [n]$  of a partial allocation as specified by a vertex  $\mathbf{x} \in V(T_{\text{half}})$ . The way these virtual valuations are defined differs based on the index  $j$ ; in particular, end bundles ( $j = 1, 4$ ) are treated differently from interior bundles ( $j = 2, 3$ ). The virtual valuations are defined as follows, for each  $\mathbf{x} \in V(T_{\text{half}})$  and each  $i \in [n]$ , where the middle row (11) applies to  $j = 2$  and  $j = 3$ :

$$\hat{u}_i(\mathbf{x}, 1) = \begin{cases} u_i(\{1, \dots, x^1 - 1\}) & \text{if } x^1 \in \mathbb{Z}, \\ u_i(\{1, \dots, x^1 - \frac{3}{2}\}) & \text{if } x^1 \notin \mathbb{Z}. \end{cases} \quad (10)$$

$$\hat{u}_i(\mathbf{x}, j) = \begin{cases} u_i^-(\{x^{j-1}, \dots, x^j\}) & \text{if } x^{j-1} \in \mathbb{Z} \text{ and } x^j \in \mathbb{Z}, \\ u_i(I^j(\mathbf{x})) & \text{otherwise.} \end{cases} \quad (11)$$

$$\hat{u}_i(\mathbf{x}, 4) = \begin{cases} u_i(\{x^3 + 1, \dots, m\}) & \text{if } x^3 \in \mathbb{Z}, \\ u_i(\{x^3 + \frac{3}{2}, \dots, m\}) & \text{if } x^3 \notin \mathbb{Z}. \end{cases} \quad (12)$$

Thus, for an interior bundle  $j = 2, 3$ , if both the items  $x^{j-1}$  and  $x^j$  to either side of the bundle are covered by a knife, an agent expects that one of these items (the less-valuable one) will be put into bundle  $I_*^j$  of the final rounded allocation (recall the definition of  $u_i^-$  in equation (1)). For exterior bundles,  $j = 1$  (resp.  $j = 4$ ), if the item  $x^1$  (resp.  $x^3$ ) is not covered by a knife, the agent does not expect the interior item (next to the knife) to belong to the final bundle  $I_*^j$ , even though it belongs to the observed bundle  $I_i^j$ . Otherwise, the virtual allocations are equal to  $u_i(I^j(\mathbf{x}))$ , so the agent expects that  $I_*^j = I_i^j$ . Later, we show that these expectations are correct up to one item.

Using these virtual valuations, we define labelling functions  $\hat{L}_i : V(T_{\text{half}}) \rightarrow [n]$  so that

$$\hat{L}_i(\mathbf{x}) \in \{j \in [n] : \hat{u}_i(\mathbf{x}, j) \geq \hat{u}_i(\mathbf{x}, k) \text{ for all } k \in [n]\}.$$

One can check that these valuation functions are still proper.

Again, by Sperner's lemma, there exists an elementary simplex  $S^* = \text{conv}(\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n)$  of the triangulation  $T_{\text{half}}$  which is fully-labeled according to our new labeling function: there is a permutation  $\phi : [n] \rightarrow [n]$ , with  $\hat{L}_i(\mathbf{x}_i) = \phi(i)$  for all  $i \in [n]$ . Again, this elementary simplex induces a Sperner sequence  $(A_1, \dots, A_n)$  of partial partitions.

To shorten a case distinction, we assume that  $y^2 \in I_1^2 \cup I_2^2 \cup I_3^2 \cup I_4^2$ , i.e., that the boundary item  $y^2$  appears in the second but not in the third bundle in the Sperner sequence. This assumption is without loss of generality, since by the left-right symmetry of the definition of virtual valuations, if necessary we can reverse the path  $P$  and consider the same elementary simplex with vertices ordered in reverse  $(\mathbf{x}_4, \mathbf{x}_3, \mathbf{x}_2, \mathbf{x}_1)$ ; it will still be fully-labeled.

With this assumption made throughout the rest of the argument, we now round the Sperner sequence into a complete partition  $(I_*^1, I_*^2, I_*^3, I_*^4)$  of  $P$  defined as follows:

$$I_*^1 := I_1^1 \cup \dots \cup I_4^1, \quad I_*^2 := I_1^2 \cup \dots \cup I_4^2, \quad I_*^3 := B^3 \cup \{y^3\}, \quad I_*^4 := B^4.$$

Depending on the placement of the boundary item  $y^1$ , we will either have  $I_*^1 = B^1$  or  $I_*^1 = B^1 \cup \{y^1\}$ ; and either  $I_*^2 = \{y^1\} \cup B^2 \cup \{y^2\}$  or  $I_*^2 = B^2 \cup \{y^2\}$ . With these choices, each interior bundle ( $j = 2, 3$ ) receives at least one of the boundary items adjacent to it.

The main part of showing that the partition  $(I_*^1, I_*^2, I_*^3, I_*^4)$  can be made into an EF1 allocation is an analogue of (9), which shows that agents' expectations about their bundle are approximately correct. The following analogous proposition is proved by case analysis.

► **Proposition 17.** *For each  $i \in [n]$  and each  $j \in [n]$ , we have  $u_i(I_*^j) \geq \hat{u}_i(\mathbf{x}_i, j) \geq u_i^-(I_*^j)$ .*

Now again, based on the partition, we can define an allocation  $A_*$  by  $A_*(i) = I_*^{\phi(i)}$  for each agent  $i \in [n]$ . Thus, each agent  $i$  receives the bundle in the complete partition corresponding to  $i$ 's most-preferred index  $\phi(i)$ . We prove that  $A_*$  satisfies EF1: For any pair  $i, j \in [n]$  of agents, we have

$$\begin{aligned} u_i(A_*(i)) = u_i(I_*^{\phi(i)}) &\geq \hat{u}_i(\mathbf{x}_i, \phi(i)) && \text{by Proposition 17} \\ &\geq \hat{u}_i(\mathbf{x}_i, \phi(j)) && \text{since } \hat{L}_i(\mathbf{x}_i) = \phi(i) \\ &\geq u_i^-(I_*^{\phi(j)}) = u_i^-(A_*(j)). && \text{by Proposition 17} \end{aligned}$$

Hence, we have proved the main result of this section:

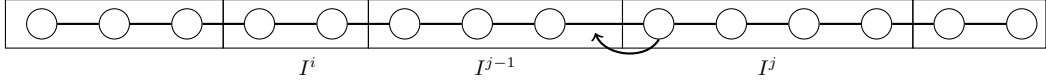
► **Theorem 18.** *On a path, for four agents with monotone valuation functions, a connected EF1 allocation exists.*

For five or more agents, we were not able to construct labeling functions and a rounding scheme which ensure that agents' expectations are correct up to one item. In the four-agent case, each interior bundle is adjacent to an exterior bundle (which helps in the construction), but for five agents, there is a middle bundle whose neighboring bundles are also interior.

## 7 EF1 existence for identical valuations

A special case of the fair division problem is the case of *identical valuations*, where all agents have the same valuation for the goods: for all agents  $i, j \in N$  and every bundle  $I \in \mathcal{C}(V)$ , we have  $u_i(I) = u_j(I)$ . We then write  $u(I)$  for the common valuation of bundle  $I$ . The case of identical valuations often allows for more positive results and an easier analysis. Indeed, we can prove that, for identical valuations and *any* number of agents, an EF1 allocation connected on a path is guaranteed to exist and can be found in polynomial time.

Now, one might guess that in the restricted case of identical valuations, egalitarian allocations are EF1. However, the leximin-optimal connected allocation may fail EF1: Consider a path with five items and additive valuations 1–3–1–1–1 shared by three agents. The unique leximin allocation is (1, 3, 1–1–1), which induces envy even up to one good. The same allocation also uniquely maximizes Nash welfare, so the Nash optimum also does not guarantee EF1. In contrast, when requiring bundles to satisfy matroid constraints (rather than connectivity constraints), the Nash optimum is EF1 with identical valuations [6].



■ **Figure 4** If  $i$  envies  $j$  even up to one good, Algorithm 1 takes an item out of bundle  $I^j$  and moves it in  $i$ 's direction.

---

**Algorithm 1** Find a connected EF1 allocation of a path  $P$  with identical valuations.

---

- 1: Let  $A = (I^1, \dots, I^n)$  be a leximin allocation of  $P$
  - 2: Fix an agent  $i$  with minimum utility in  $A$ , i.e.,  $u(I^i) \leq u(I^j)$  for all  $j \in [n]$
  - 3: **for**  $j = 1, \dots, i - 1$  **do**
  - 4:     **if**  $i$  envies  $I^j$  even up to one good, i.e.,  $u(I^i) < u^-(I^j)$  **then**
  - 5:         repeatedly delete the right-most item of  $I^j$  and add it to  $I^{j+1}$  until  $u(I^i) \geq u^-(I^j)$
  - 6: **for**  $j = n, \dots, i + 1$  **do**
  - 7:     **if**  $i$  envies  $I^j$  even up to one good, i.e.,  $u(I^i) < u^-(I^j)$  **then**
  - 8:         repeatedly delete the left-most item of  $I^j$  and add it to  $I^{j-1}$  until  $u(I^i) \geq u^-(I^j)$
  - 9: **return**  $A$
- 

Maximizing an egalitarian objective seemed promising because it ensures that no-one is too badly off, and therefore has not much reason to envy others. The problem is that some bundles might be too desirable. To fix this, we could try to reallocate items so that no bundle is too valuable. This is exactly the strategy of our algorithm: It starts with a leximin allocation, and then moves items from high-value bundles to lower-value bundles, until the result is EF1. In more detail, the algorithm identifies one agent  $i$  who is worst-off in the leximin allocation, and then adjusts the allocation so that  $i$  does not envy any other bundle up to one good. The algorithm does this by going through all bundles in the allocation, outside-in, and if  $i$  envies a bundle  $I^j$  even up to one good, it moves one item from  $I^j$  inwards (in  $i$ 's direction), see Figure 4. As we will show, a key invariant preserved by the algorithm is that the value of  $I^i$  never increases, and  $i$  remains worst-off. Thus, since  $i$  does not envy others up to one good, the allocation at the end is EF1.

Formally, a *leximin allocation* is an allocation which maximizes the lowest utility of an agent; subject to that it maximizes the second-lowest utility, and so on. In particular, if the highest achievable minimum utility is  $u_L$ , then the leximin allocation is such that every agent has utility at least  $u_L$ , and the number of agents with utility exactly  $u_L$  is minimum.

► **Theorem 19.** *For identical valuations on a path, Algorithm 1 finds an EF1 allocation.*

**Proof.** For an allocation  $A = (I^1, \dots, I^n)$ , write  $u_L(A) := \min_{j \in [n]} u(I^j)$  for the minimum utility obtained in  $A$ , and write  $L(A) := \{j \in [n] : u(I^j) = u_L(A)\}$  for the set of agents (*losers*) who obtain this utility. For the leximin allocation  $A_{\text{leximin}}$  obtained at the start of the algorithm, write  $u_L^* := u_L(A_{\text{leximin}})$  and  $L^* := L(A_{\text{leximin}})$ . Note that by leximin-optimality, for every allocation  $A$  we must have  $u_L(A) \leq u_L^*$ , and if  $u_L(A) = u_L^*$  then  $|L(A)| \geq |L^*|$ . Let  $i \in L^*$  be the agent fixed at the start of the algorithm.

*Claim 1.* Throughout the algorithm,  $u_L(A) = u_L^*$  and  $L(A) = L^*$ .

The claim is true before we start the for-loops. Suppose the claim holds up until some iteration of the first for-loop, and we now move an item from  $I^j$  to  $I^{j+1}$ , obtaining the new bundles  $I_{\text{new}}^j$  and  $I_{\text{new}}^{j+1}$  in the new allocation  $A_{\text{new}}$ . Then  $u(I_{\text{new}}^j) \geq u^-(I^j) > u(I^i) = u_L^*$ , where the strict inequality holds by the if- and until-clauses. Since no agent other than  $j$  has become worse-off in  $A_{\text{new}}$ , it follows that  $u_L(A_{\text{new}}) \geq u_L(A) = u_L^*$ . As noted, by optimality of  $u_L^*$ , we have  $u_L(A_{\text{new}}) \leq u_L^*$ . Hence  $u_L(A_{\text{new}}) = u_L^*$ . Thus, by optimality of

$L^*$ , we have  $|L(A_{\text{new}})| \geq |L^*|$ . Because agent  $j$  has not become a loser (since  $u(I_{\text{new}}^j) > u_L^*$ , as shown before) and no other agent has become a loser, we have  $L(A_{\text{new}}) \subseteq L(A) = L^*$ . Thus  $L(A_{\text{new}}) = L^*$ , as required. The second for-loop is handled similarly.

*Claim 2.* After both for-loops terminate, agent  $i$  does not envy any agent up to one good.

For any  $j \neq i$ , agent  $i$  does not envy  $j$  up to one good immediately after the relevant loop has handled  $j$ , and at no later stage of the algorithm does  $I^j$  change.

It follows that the allocation  $A$  returned by the algorithm is EF1: By Claim 1, we have  $i \in L(A)$ , so that  $u(I^j) \geq u(I^i)$  for all  $j \in [n]$ . By Claim 2, agent  $i$  does not envy any other agent up to one good, so that  $u(I^i) \geq u^-(I^k)$  for all  $k \in [n]$ . Hence, for all  $j, k \in [n]$ , we have  $u(I^j) \geq u^-(I^k)$ , that is, no agent envies another agent up to one good. ◀

Algorithm 1 can be implemented to run in polynomial time, because with identical valuations, one can use dynamic programming to find a leximin allocation in time  $O(m^2n^2)$ , and the remainder of Algorithm 1 takes time  $O(mn)$ , as each item is moved at most  $n$  times.

The reallocation stage of our algorithm bears some similarity to Suksompong's [30, Thm. 2] proof that a  $u_{\max}$ -equitable allocation exists. In a very recent paper, Oh et al. [23, Lem. C.2] proved independently, using an inductive argument, that EF1 allocations on a path exist for identical valuations, and can be found in polynomial time.

## 8 Conclusion

We have studied the existence of EF1 allocations under connectivity constraints imposed by an undirected graph. We have shown that for two, three, or four agents, an EF1 allocation exists if the graph is traceable. For any number of agents, we also proved that traceable graphs guarantee the existence of an EF2 allocation, and they guarantee the existence of an EF1 allocation with identical valuations.

Several questions are open. We did not settle whether EF1 allocations on a path exist for five or more agents. Our Sperner technique for four agents seems to not extend to five agents. Extensive sampling did not find an example where no EF1 allocation exists.

Many of our procedures admit efficient implementations for finding fair allocations, but for our Sperner results we do not know better algorithms than searching through all connected allocations. For divisible cake-cutting, it is PPAD-complete to find an envy-free allocation [14]. What is the complexity of finding an EF1 or EF2 allocation on a path?

In the setting without connectivity constraints, it is possible to achieve efficiency and fairness simultaneously: the maximum Nash welfare solution yields an allocation that is both EF1 and Pareto-optimal [13, 4]. In our model, this is unfortunately impossible, since on a path there are instances where there is no connected allocation which is EF1 and Pareto-optimal [18], and it is NP-hard to decide whether such an allocation exists [18].

In this paper, we have only considered *goods*, with monotonic valuations. The setting where some or all items are undesirable (so-called *chores*) is also of interest [7, 22, 27, 1]. On a path, a connected *Prop1* allocation always exists [1], but the existence of EF1 or EF2 allocations in this domain is open. For cake-cutting, when agents consider some parts of the cake undesirable, Sperner's lemma does not directly produce a connected envy-free allocation [27], but other methods can prove the existence of such allocations in most cases [27, 22].

---

## References

- 1 H. Aziz, I. Caragiannis, and A. Igarashi. Fair allocation of combinations of indivisible goods and chores. *CoRR*, 2018. arXiv:1807.10684.
- 2 R. B.apat. A constructive proof of a permutation-based generalization of Sperner's Lemma. *Mathematical Programming*, 44(1):113–120, 1989.

- 3 J. B. Barbanel and S. J. Brams. Cake division with minimal cuts: envy-free procedures for three persons, four persons, and beyond. *Mathematical Social Sciences*, 48(3):251–269, 2004.
- 4 S. Barman, S. K. Krishnamurthy, and R. Vaish. Finding fair and efficient allocations. In *Proceedings of the 19th ACM Conference on Economics and Computation (ACM-EC)*, pages 557–574, 2018.
- 5 V. Bilò, I. Caragiannis, M. Flammini, A. Igarashi, G. Monaco, D. Peters, C. Vinci, and W. S. Zwicker. Almost Envy-Free Allocations with Connected Bundles. *CoRR*, 2018. arXiv:1808.09406.
- 6 A. Biswas and S. Barman. Fair division under cardinality constraints. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence (IJCAI)*, pages 91–97, 2018.
- 7 A. Bogomolnaia, H. Moulin, F. Sandomirskiy, and E. Yanovskaya. Dividing goods and bads under additive utilities. *CoRR*, 2016. arXiv:1610.03745.
- 8 J. A. Bondy and U. S. R. Murty. *Graph Theory*. Springer, 1st edition, 2008.
- 9 S. Bouveret, K. Cechlárová, E. Elkind, A. Igarashi, and D. Peters. Fair division of a graph. In *Proceedings of the 24th International Joint Conference on Artificial Intelligence (IJCAI)*, pages 135–141, 2017.
- 10 S. J. Brams and A. D. Taylor. *Fair division: from cake-cutting to dispute resolution*. Cambridge University Press, 1996.
- 11 S. J. Brams, A. D. Taylor, and W. S. Zwicker. A moving-knife solution to the four-person envy-free cake-division problem. *Proceedings of the American Mathematical Society*, 125(2):547–554, 1997.
- 12 E. Budish. The combinatorial assignment problem: Approximate competitive equilibrium from equal incomes. *Journal of Political Economy*, 119(6):1061–1103, 2011.
- 13 I. Caragiannis, D. Kurokawa, H. Moulin, A. D. Procaccia, N. Shah, and J. Wang. The unreasonable fairness of maximum Nash welfare. In *Proceedings of the 17th ACM Conference on Economics and Computation (ACM-EC)*, pages 305–322, 2016.
- 14 X. Deng, Q. Qi, and A. Saberi. Algorithmic solutions for envy-free cake cutting. *Operations Research*, 60(6):1461–1476, 2012.
- 15 L. E. Dubins and E. H. Spanier. How to cut a cake fairly. *The American Mathematical Monthly*, 68(1):1–17, 1961.
- 16 S. Even and R. E. Tarjan. Computing an st-numbering. *Theoretical Computer Science*, 2(3):339–344, 1976.
- 17 J. Hopcroft and R. Tarjan. Algorithm 447: efficient algorithms for graph manipulation. *Communications of the ACM*, 16(6):372–378, 1973.
- 18 A. Igarashi and D. Peters. Pareto-optimal allocation of indivisible goods with connectivity constraints. In *Proceedings of the 33rd AAAI Conference on Artificial Intelligence (AAAI)*, 2019. arXiv:1811.04872.
- 19 H. W. Kuhn. Some combinatorial lemmas in topology. *IBM Journal of Research and Development*, 4(5):518–524, 1960.
- 20 A. Lempel, S. Even, and I. Cederbaum. An algorithm for planarity testing of graphs. *Theory of Graphs: International Symposium*, pages 215–232, 1967.
- 21 R. J. Lipton, E. Markakis, E. Mossel, and A. Saberi. On approximately fair allocations of indivisible goods. In *Proceedings of the 5th ACM Conference on Electronic Commerce (ACM-EC)*, pages 125–131, 2004.
- 22 F. Meunier and S. Zerbib. Envy-free divisions of a partially burnt cake. *CoRR*, 2018. arXiv:1804.00449.
- 23 H. Oh, A. D. Procaccia, and W. Suksompong. Fairly Allocating Many Goods with Few Queries. In *Proceedings of the 33rd AAAI Conference on Artificial Intelligence (AAAI)*, 2019. arXiv:1807.11367.



- 24 A. D. Procaccia. Cake Cutting Algorithms. In F. Brandt, V. Conitzer, U. Endriss, J. Lang, and A. D. Procaccia, editors, *Handbook of Computational Social Choice*, chapter 13. Cambridge University Press, 2016.
- 25 J. Robertson and W. Webb. *Cake-cutting algorithms: Be fair if you can*. AK Peters/CRC Press, 1998.
- 26 H. E. Scarf. The computation of equilibrium prices: an exposition. *Handbook of Mathematical Economics*, 2:1007–1061, 1982.
- 27 E. Segal-Halevi. Fairly dividing a cake after some parts were burnt in the oven. In *Proceedings of the 17th International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, pages 1276–1284, 2018. arXiv:1704.00726.
- 28 W. Stromquist. How to cut a cake fairly. *The American Mathematical Monthly*, 87(8):640–644, 1980.
- 29 F. E. Su. Rental harmony: Sperner’s lemma in fair division. *The American Mathematical Monthly*, 106(10):930–942, 1999.
- 30 W. Suksompong. Fairly allocating contiguous blocks of indivisible items. In *Proceedings of the 10th International Symposium on Algorithmic Game Theory (SAGT)*, pages 333–344, 2017. arXiv:1707.00345.
- 31 R. E. Tarjan. Two streamlined depth-first search algorithms. *CoRR*, 9:85–94, 1986.




# “Quantum Supremacy” and the Complexity of Random Circuit Sampling

## Adam Bouland

Electrical Engineering and Computer Sciences, University of California, Berkeley, 387 Soda Hall Berkeley, CA 94720, U.S.A.

abouland@berkeley.edu


 <https://orcid.org/0000-0002-8556-8337>

## Bill Fefferman

Electrical Engineering and Computer Sciences, University of California, Berkeley, 387 Soda Hall Berkeley, CA 94720, U.S.A.

Joint Center for Quantum Information and Computer Science (QuICS), University of Maryland/NIST, Bldg 224 Stadium Dr Room 3100, College Park, MD 20742, U.S.A.


wjf@umiacs.umd.edu

 <https://orcid.org/0000-0002-9627-0210>

## Chinmay Nirkhe

Electrical Engineering and Computer Sciences, University of California, Berkeley, 387 Soda Hall Berkeley, CA 94720, U.S.A.

nirkhe@cs.berkeley.edu

 <https://orcid.org/0000-0002-5808-4994>

## Umesh Vazirani

Electrical Engineering and Computer Sciences, University of California, Berkeley, 387 Soda Hall Berkeley, CA 94720, U.S.A.

vazirani@cs.berkeley.edu

---

### Abstract

---

A critical goal for the field of quantum computation is quantum supremacy – a demonstration of any quantum computation that is prohibitively hard for classical computers. It is both a necessary milestone on the path to useful quantum computers as well as a test of quantum theory in the realm of high complexity. A leading near-term candidate, put forth by the Google/UCSB team, is sampling from the probability distributions of randomly chosen quantum circuits, called Random Circuit Sampling (RCS).

While RCS was defined with experimental realization in mind, we give strong complexity-theoretic evidence for the classical hardness of RCS, placing it on par with the best theoretical proposals for supremacy. Specifically, we show that RCS satisfies an average-case hardness condition – computing output probabilities of typical quantum circuits is as hard as computing them in the worst-case, and therefore  $\#P$ -hard. Our reduction exploits the polynomial structure in the output amplitudes of random quantum circuits, enabled by the Feynman path integral. In addition, it follows from known results that RCS also satisfies an anti-concentration property, namely that errors in estimating output probabilities are small with respect to the probabilities themselves. This makes RCS the first proposal for quantum supremacy with both of these properties. We also give a natural condition under which an existing statistical measure, cross-entropy, verifies RCS, as well as describe a new verification measure which in some formal sense maximizes the information gained from experimental samples.

**2012 ACM Subject Classification** Theory of computation → Quantum complexity theory

**Keywords and phrases** quantum supremacy, average-case hardness, verification

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2019.15



© Adam Bouland, Bill Fefferman, Chinmay Nirkhe, and Umesh Vazirani; licensed under Creative Commons License CC-BY

10th Innovations in Theoretical Computer Science (ITCS 2019).

Editor: Avrim Blum; Article No. 15; pp. 15:1–15:2

Leibniz International Proceedings in Informatics



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

## 15:2 “Quantum Supremacy” and the Complexity of Random Circuit Sampling

**Related Version** A full version of this paper, including all proofs, is available at [1], <https://doi.org/10.1038/s41567-018-0318-2>.

**Funding** A.B., B.F., C.N. and U.V. were supported by ARO grant W911NF-12-1-0541 and NSF grant CCF-1410022 and a Vannevar Bush faculty fellowship. B.F. is supported in part by an Air Force Office of Scientific Research Young Investigator Program award number FA9550-18-1-0148. Parts of this work were done at the Kavli Institute for Theoretical Physics. Portions of this paper are a contribution of NIST, an agency of the US government, and are not subject to US copyright.

**Acknowledgements** We thank S. Aaronson, D. Aharonov, F. Brandão, M. Coudron, A. Deshpande, T. Gur, Z. Landau, N. Spooner and H. Yuen for helpful discussions.

---

### References

- 1 Adam Bouland, Bill Fefferman, Chinmay Nirkhe, and Umesh Vazirani. On the complexity and verification of quantum random circuit sampling. *Nature Physics*, 2018. doi:10.1038/s41567-018-0318-2.

# Adversarially Robust Property-Preserving Hash Functions

Elette Boyle<sup>1</sup>

IDC Herzliya, Kanfei Nesharim Herzliya, Israel  
elette.boyle@idc.ac.il

Rio LaVigne<sup>2</sup>

MIT CSAIL, 32 Vassar Street, Cambridge MA, 02139 USA  
rio@mit.edu

Vinod Vaikuntanathan<sup>3</sup>

MIT CSAIL, 32 Vassar Street, Cambridge MA, 02139 USA  
vinodv@mit.edu

---

## Abstract

Property-preserving hashing is a method of compressing a large input  $\mathbf{x}$  into a short hash  $h(\mathbf{x})$  in such a way that given  $h(\mathbf{x})$  and  $h(\mathbf{y})$ , one can compute a property  $P(\mathbf{x}, \mathbf{y})$  of the original inputs. The idea of property-preserving hash functions underlies sketching, compressed sensing and locality-sensitive hashing.

Property-preserving hash functions are usually probabilistic: they use the random choice of a hash function from a family to achieve compression, and as a consequence, err on some inputs. Traditionally, the notion of correctness for these hash functions requires that for every two inputs  $\mathbf{x}$  and  $\mathbf{y}$ , the probability that  $h(\mathbf{x})$  and  $h(\mathbf{y})$  mislead us into a wrong prediction of  $P(\mathbf{x}, \mathbf{y})$  is negligible. As observed in many recent works (incl. Mironov, Naor and Segev, STOC 2008; Hardt and Woodruff, STOC 2013; Naor and Yogev, CRYPTO 2015), such a correctness guarantee assumes that the adversary (who produces the offending inputs) has no information about the hash function, and is too weak in many scenarios.

We initiate the study of *adversarial robustness* for property-preserving hash functions, provide definitions, derive broad lower bounds due to a simple connection with communication complexity, and show the necessity of computational assumptions to construct such functions. Our main positive results are two candidate constructions of property-preserving hash functions (achieving different parameters) for the (promise) gap-Hamming property which checks if  $\mathbf{x}$  and  $\mathbf{y}$  are “too far” or “too close”. Our first construction relies on generic collision-resistant hash functions, and our second on a variant of the syndrome decoding assumption on low-density parity check codes.

**2012 ACM Subject Classification** Theory of computation → Cryptographic primitives

**Keywords and phrases** Hash function, compression, property-preserving, one-way communication

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2019.16

---

<sup>1</sup> Supported in part by ISF grant 1861/16, AFOSR Award FA9550-17-1-0069, and ERC grant 742754 (project NTSC).

<sup>2</sup> This material is based upon work supported by the National Science Foundation Graduate Research Fellowship under Grant No. 1122374. Any opinion, findings, and conclusions or recommendations expressed in this material are those of the authors(s) and do not necessarily reflect the views of the National Science Foundation. This research was also supported in part by grants of the third author.

<sup>3</sup> Research supported in part by NSF Grants CNS-1350619 and CNS-1414119, an MIT-IBM grant and a DARPA Young Faculty Award.



© Elette Boyle, Rio LaVigne, and Vinod Vaikuntanathan;  
licensed under Creative Commons License CC-BY

10th Innovations in Theoretical Computer Science (ITCS 2019).

Editor: Avrim Blum; Article No. 16; pp. 16:1–16:20

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://eprint.iacr.org/2018/1158.pdf>.

**Acknowledgements** We thank Daniel Wichs for suggesting the construction in Section 4 and for discussions regarding the connection to secure sketches and fuzzy extractors, and Shafi Goldwasser for enlightening conversations on the connections to robustness in machine learning.

## 1 Introduction

The problem of *property-preserving hashing*, namely how to compress a large input in a way that preserves a class of its properties, is an important one in the modern age of massive data. In particular, the idea of property-preserving hashing underlies sketching [23, 22, 1, 8, 6], compressed sensing [7], locality-sensitive hashing [14], and in a broad sense, much of machine learning.

As two concrete examples in theoretical computer science, consider universal hash functions [5] which can be used to test the equality of data points, and locality-sensitive hash functions [14, 13] which can be used to test the  $\ell_p$ -distance between vectors. In both cases, we trade off accuracy in exchange for compression. For example, in the use of universal hash functions to test for equality of data points, one stores the hash  $h(x)$  of a point  $x$  together with the description of the hash function  $h$ . Later, upon obtaining a point  $y$ , one computes  $h(y)$  and checks if  $h(y) = h(x)$ . The pigeonhole principle tells us that mistakes are inevitable; all one can guarantee is that they happen with an acceptably small probability. More precisely, universal hash functions tell us that

$$\forall x \neq y \in D, \Pr[h \leftarrow \mathcal{H} : h(x) \neq h(y)] \geq 1 - \epsilon$$

for some small  $\epsilon$ . A cryptographer's way of looking at such a statement is that it asks the adversary to pick  $x$  and  $y$  first; and evaluates her success w.r.t. a hash function chosen randomly from the family  $\mathcal{H}$ . In particular, the adversary has no information about the hash function when she comes up with the (potentially) offending inputs  $x$  and  $y$ . Locality-sensitive hash functions have a similar flavor of correctness guarantee.

The starting point of this work is that this definition of correctness is too weak in the face of adversaries with access to the hash function (either the description of the function itself or perhaps simply oracle access to its evaluation). Indeed, in the context of equality testing, we have by now developed several notions of robustness against such adversaries, in the form of pseudorandom functions (PRF) [11], universal one-way hash functions (UOWHF) [25] and collision-resistant hash functions (CRHF). Our goal in this work is to expand the reach of these notions beyond testing equality; that is, our aim is *to do unto property-preserving hashing what CRHFs did to universal hashing*.

Several works have observed the deficiency of the universal hash-type definition in adversarial settings, including a wide range of recent attacks within machine learning in adversarial environments (e.g., [20, 17, 28, 26, 16]). Such findings motivate a rigorous approach to combatting adversarial behavior in these settings, a direction in which significantly less progress has been made. Mironov, Naor and Segev [21] showed *interactive protocols* for sketching in such an adversarial environment; in contrast, we focus on non-interactive hash functions. Hardt and Woodruff [12] showed negative results which say that linear functions cannot be robust (even against computationally bounded adversaries) for certain natural  $\ell_p$  distance properties; our work will use non-linearity and computational assumptions to overcome the [12] attack. Finally, Naor and Yagev [24] study adversarial Bloom filters

which compress a set in a way that supports checking set membership; we were able to use their lower bound techniques (in the full version of this paper), proving the necessity for cryptographic assumptions for many predicates.

### Motivating Robustness: Facial Recognition

In the context of facial recognition, authorities  $A$  and  $B$  store the captured images  $x$  of suspects. At various points in time, say authority  $A$  wishes to look up  $B$ 's database for a suspect with face  $x$ .  $A$  can do so by comparing  $h(x)$  with  $h(y)$  for all  $y$  in  $B$ 's database.

This application scenario motivated prior notions of fuzzy extractors and secure sketching. As with secure sketches and fuzzy extractors, a locality-sensitive property-preserving hash guarantees that close inputs (facial images) remain close when hashed [9]; this ensures that small changes in one's appearance do not affect whether or not that person is authenticated. However, neither fuzzy extractors nor secure sketching guarantees that *far* inputs remain far when hashed. Consider an adversarial setting, not where a person wishes to evade detection, but where she wishes to be mistaken for someone else. Her face  $x'$  will undoubtedly be different (far) from her target  $x$ , but there is nothing preventing her from slightly altering her face and passing as a completely different person when using a system with such a one-sided guarantee. This is where our notion of robustness comes in (as well as the need for cryptography): not only will adversarially chosen close  $x$  and  $x'$  map to close  $h(x)$  and  $h(x')$ , but if adversarially chosen  $x$  and  $x'$  are *far*, they will be mapped to far outputs, unless the adversary is able to break a cryptographic assumption.

### Comparison to Secure Sketches and Fuzzy Extractors

It is worth explicitly comparing fuzzy extractors and secure sketching to this primitive [9], as they aim to achieve similar goals. Both of these seek to preserve the privacy of their inputs. Secure sketches generate random-looking sketches that hide information about the original input so that the original input can be reconstructed when given something close to it. Fuzzy extractors generate uniform-looking keys based off of fuzzy (biometric) data also using entropy: as long as the input has enough entropy, so will the output. As stated above, both guarantee that if inputs are close, they will 'sketch' or 'extract' to the same object. Now, the entropy of the sketch or key guarantees that randomly generated far inputs will not collide, but there are no guarantees about adversarially generated far inputs. To use the example above, it could be that once an adversary sees a sketch or representation, she can generate two far inputs that will reconstruct to the correct input.

### Robust Property-Preserving Hash Functions

We put forth several notions of *robustness* for property-preserving hash (PPH) functions which capture adversaries with increasing power and access to the hash function. We then ask which properties admit robust property-preserving hash functions, and show positive and negative results.

- On the negative side, using a connection to communication complexity, we show that most properties and even simple ones such as set disjointness, inner product and greater-than do not admit non-trivial property-preserving hash functions.
- On the positive side, we provide two constructions of robust property-preserving hash functions (satisfying the strongest of our notions). The first is based on the standard cryptographic assumption of collision-resistant hash functions, and the second achieves

more aggressive parameters under a new assumption related to the hardness of syndrome decoding on low density parity-check (LDPC) codes. The first is expanded upon in this version (section 4), while the second is in the full version.

- Finally, in the full version, we show that for essentially any non-trivial predicate (which we call collision-sensitive), achieving even a mild form of robustness requires cryptographic assumptions.

We proceed to describe our contributions in more detail.

## 1.1 Our Results and Techniques

We explore two notions of properties. The first is that of property classes  $\mathcal{P} = \{P : D \rightarrow \{0, 1\}\}$ , sets of single-input predicates. This notion is the most general, and is the one in which we prove lower bounds. The second is that of two-input properties  $P : D \times D \rightarrow \{0, 1\}$ , which compares two inputs. This second notion is more similar to standard notions of universal hashing and collision-resistance, stronger than the first, and where we get our constructions. We note that a two-input predicate has an analogous predicate-class  $\mathcal{P} = \{P_x\}_{x \in D}$ , where  $P_{x_1}(x_2) = P(x_1, x_2)$ .

The notion of a property can be generalized in many ways, allowing for promise properties which output 0, 1 or  $\star$  (a don't care symbol), and allowing for more than 2 inputs. The simplest notion of correctness for property-preserving hash functions requires that, analogously to universal hash functions,

$$\forall x, y \in D, \Pr[h \leftarrow \mathcal{H} : \mathcal{H}.\text{Eval}(h, h(x), h(y)) \neq P(x, y)] = \text{negl}(\lambda)$$

or for single-input predicate-classes

$$\forall x \in D \text{ and } P \in \mathcal{P}, \Pr[h \leftarrow \mathcal{H} : \mathcal{H}.\text{Eval}(h, h(x), P) \neq P(x)] = \text{negl}(\lambda)$$

where  $\lambda$  is a security parameter. Notice in the single-input case, the “second” input can be seen as the predicate chosen from the class.

For the sake of simplicity in our overview, we will focus on two-input predicates.

### Defining Robust Property-Preserving Hashing

We define several notions of *robustness* for PPH, each one stronger than the last. Here, we describe the strongest of all, called *direct-access PPH*.

In a direct-access PPH, the (polynomial-time) adversary is given the hash function and is asked to find a pair of bad inputs, namely  $x, y \in D$  such that  $\mathcal{H}.\text{Eval}(h, h(x), h(y)) \neq P(x, y)$ . That is, we require that

$$\forall \text{ p.p.t. } \mathcal{A}, \Pr[h \leftarrow \mathcal{H}; (x, y) \leftarrow \mathcal{A}(h) : \mathcal{H}.\text{Eval}(h, h(x), h(y)) \neq P(x, y)] = \text{negl}(\lambda),$$

where we use the notation  $\Pr[A_1; \dots; A_m : E]$  to denote the probability that event  $E$  occurs following an experiment defined by executing the sequence  $A_1, \dots, A_m$  in order. The direct-access definition is the analog of collision-resistant hashing for general properties.

Our other definitions vary by how much access the adversary is given to the hash function, and are motivated by different application scenarios. From the strong to weak, these include double-oracle PPH where the adversary is given access to a hash oracle and a hash evaluation oracle, and evaluation-oracle PPH where the adversary is given only a combined oracle. Definitions similar to double-oracle PPH have been proposed in the context of adversarial bloom filters [24], and ones similar to evaluation-oracle PPH have been proposed in the context of showing attacks against property-preserving hash functions [12]. For more details, we refer the reader to Section 2.



## Connections to Communication Complexity and Negative Results

Property-preserving hash functions for a property  $P$ , even without robustness, imply communication-efficient protocols for  $P$  in several models. For example, any PPH for  $P$  implies a protocol for  $P$  in the simultaneous messages model of Babai, Gal, Kimmel and Lokam [3] wherein Alice and Bob share a common random string  $h$ , and hold inputs  $x$  and  $y$  respectively. Their goal is to send a single message to Charlie who should be able to compute  $P(x, y)$  except with small error. Similarly, another formalization of PPH that we present, called PPH for single-input predicate classes (see Section 2) implies efficient protocols in the one-way communication model [31].

We use known lower bounds in these communication models to rule out PPHs for several interesting predicates (even without robustness). There are two major differences between the PPH setting and the communication setting, however: (a) in the PPH setting, we demand an error that is negligible (in a security parameter); and (b) we are happy with protocols that communicate  $n - 1$  bits (or the equivalent bound in the case of promise properties) whereas the communication lower bounds typically come in the form of  $\Omega(n)$  bits. In other words, the communication lower bounds *as-is* do not rule out PPH.

At first thought, one might be tempted to think that the negligible-error setting is the same as the deterministic setting where there are typically lower bounds of  $n$  (and not just  $\Omega(n)$ ); however, this is not the case. For example, the equality function which has a negligible error public-coin simultaneous messages protocol (simply using universal hashing) with communication complexity  $CC = O(\lambda)$  and deterministic protocols require  $CC \geq n$ . Thus, deterministic lower bounds do not (indeed, cannot) do the job, and we must better analyze the randomized lower bounds. Our refined analysis shows the following lower bounds:

- PPH for the Gap-Hamming (promise) predicate with a gap of  $\sqrt{n}/2$  is impossible by refining the analysis of a proof by Jayram, Kumar and Sivakumar [15]. The Gap-Hamming predicate takes two vectors in  $\{0, 1\}^n$  as input, outputs 1 if the vectors are very far, 0 if they are very close, and we do not care what it outputs for inputs in the middle.
- We provide a framework for proving PPHs are impossible for some total predicates, characterizing these classes as *reconstructing*. A predicate-class is *reconstructing* if, when only given oracle access to the predicates of a certain value  $x$ , we can efficiently determine  $x$  with all but negligible probability.<sup>4</sup> With this framework, we show that PPH for the Greater-Than (GT) function is impossible. It was known that GT required  $\Omega(n)$  bits (for constant error) [27], but we show a lower bound of exactly  $n$  if we want negligible error. Index and Exact-Hamming are also reconstructing predicates.
- We also obtain a lower bound for a variant of GT: the (promise) Gap- $k$  GT predicate which on inputs  $x, y \in [N = 2^n]$ , outputs 1 if  $x - y > k$ , 0 if  $y - x > k$ , and we do not care what it outputs for inputs in between. Here, exactly  $n - \log(k) - 1$  bits are required for a perfect PPH. This is tight: we show that with fewer bits, one cannot even have a non-robust PPH, whereas there is a perfect robust PPH that compresses to  $n - \log(k) - 1$  bits.

## New Constructions

Our positive results are two constructions of a direct-access PPH for gap-Hamming for  $n$ -length vectors for large gaps of the form  $\sim O(n/\log n)$  (as opposed to an  $O(\sqrt{n})$ -gap for which we have a lower bound). Let us recall the setting: the gap Hamming predicate  $P_{\text{ham}}$ ,

<sup>4</sup> In the single-predicate language of above, the predicate class corresponds to  $\mathcal{P} = \{P(x, \cdot)\}$ .

parameterized by  $n, d$  and  $\epsilon$ , takes as input two  $n$ -bit vectors  $x$  and  $y$ , and outputs 1 if the Hamming distance between  $x$  and  $y$  is greater than  $d(1 + \epsilon)$ , 0 if it is smaller than  $d(1 - \epsilon)$  and a don't care symbol  $\star$  otherwise. To construct a direct-access PPH for this (promise) predicate, one has to construct a compressing family of functions  $\mathcal{H}$  such that

$$\begin{aligned} \forall \text{ p.p.t. } \mathcal{A}, \Pr[h \leftarrow \mathcal{H}; (x, y) \leftarrow \mathcal{A}(h) : P_{\text{ham}}(x, y) \neq \star \\ \wedge \mathcal{H}.\text{Eval}(h, h(x), h(y)) \neq P_{\text{ham}}(x, y)] = \text{negl}(\lambda). \end{aligned} \quad (1)$$

Our two constructions offer different benefits. The first provides a clean general approach, and relies on the standard cryptographic assumption of collision-resistant hash functions. The second builds atop an existing one-way communication protocol, supports a smaller gap and better efficiency, and ultimately relies on a (new) variant of the syndrome decoding assumption on low-density parity check codes.

**Construction 1.** The core idea of the first construction is to reduce the goal of robust Hamming PPH to the simpler one of robust equality testing; or, in a word, “subsampling.” The intuition is to notice that if  $\mathbf{x}_1 \in \{0, 1\}^n$  and  $\mathbf{x}_2 \in \{0, 1\}^n$  are *close*, then *most small enough subsets* of indices of  $\mathbf{x}_1$  and  $\mathbf{x}_2$  will match identically. On the other hand, if  $\mathbf{x}_1$  and  $\mathbf{x}_2$  are *far*, then *most large enough subsets* of indices will differ.

The hash function construction will thus fix a collection of sets  $\mathcal{S} = \{S_1, \dots, S_k\}$ , where each  $S_i \subseteq [n]$  is a subset of appropriately chosen size  $s$ . The desired structure can be achieved by defining the subsets  $S_i$  as the neighbor sets of a bipartite expander. On input  $\mathbf{x} \in \{0, 1\}^n$ , the hash function will consider the vector  $\mathbf{y} = (\mathbf{x}|_{S_1}, \dots, \mathbf{x}|_{S_k})$  where  $\mathbf{x}|_S$  denotes the substring of  $\mathbf{x}$  indexed by the set  $S$ . The observation above tells us that if  $\mathbf{x}_1$  and  $\mathbf{x}_2$  are close (resp. far), then so are  $\mathbf{y}_1$  and  $\mathbf{y}_2$ .

Up to now, it is not clear that progress has been made: indeed, the vector  $\mathbf{y}$  is not compressing (in which case, why not stick with  $\mathbf{x}_1, \mathbf{x}_2$  themselves?). However,  $\mathbf{y}_1, \mathbf{y}_2$  satisfy the desired Hamming distance properties with fewer symbols over a *large alphabet*,  $\{0, 1\}^s$ . As a final step, we can then leverage (standard) collision-resistant hash functions (CRHF) to compress these symbols. Namely, the final output of our hash function  $h(\mathbf{x})$  will be the vector  $(g(\mathbf{x}|_{S_1}), \dots, g(\mathbf{x}|_{S_k}))$ , where each substring of  $\mathbf{x}$  is individually compressed by a CRHF  $g$ .

The analysis of the combined hash construction then follows cleanly via two steps. The (computational) collision-resistance property of  $g$  guarantees that any efficiently found pair of inputs  $\mathbf{x}_1, \mathbf{x}_2$  will satisfy that their hash outputs

$$h(\mathbf{x}_1) = (g(\mathbf{x}_1|_{S_1}), \dots, g(\mathbf{x}_1|_{S_k})) \quad \text{and} \quad h(\mathbf{x}_2) = (g(\mathbf{x}_2|_{S_1}), \dots, g(\mathbf{x}_2|_{S_k}))$$

are close if and only if it holds that

$$(\mathbf{x}_1|_{S_1}, \dots, \mathbf{x}_1|_{S_k}) \quad \text{and} \quad (\mathbf{x}_2|_{S_1}, \dots, \mathbf{x}_2|_{S_k})$$

are close as well; that is,  $\mathbf{x}_1|_{S_i} = \mathbf{x}_2|_{S_i}$  for most  $S_i$ . (Anything to the contrary would imply finding a collision in  $g$ .) Then, the combinatorial properties of the chosen index subsets  $S_i$  ensures (unconditionally) that any such inputs  $\mathbf{x}_1, \mathbf{x}_2$  must themselves be close. The remainder of the work is to specify appropriate parameter regimes for which the CRHF can be used and the necessary bipartite expander graphs exist. Informally, we get the following theorem statement:

► **Theorem 1** (Collision-resistance-PPH informal). *Let  $\lambda$  be a security parameter. Assuming that CRHFs exist, for any polynomial  $n = n(\lambda)$ , and any constants  $\epsilon, \eta, c > 0$ , there is an  $\eta$ -compressing robust property preserving hash family for  $\text{GAPHAMMING}(n, d, \epsilon)$  where  $d = o(n/\lambda^c)$ .*

**Construction 2.** The starting point for our second construction is a simple non-robust hash function derived from a one-way communication protocol for gap-Hamming due to Kushilevitz, Ostrovsky, and Rabani [19]. In a nutshell, the hash function is parameterized by a random sparse  $m \times n$  matrix  $A$  with 1's in  $1/d$  of its entries and 0's elsewhere; multiplying this matrix by a vector  $\mathbf{z}$  “captures” information about the Hamming weight of  $\mathbf{z}$ . However, this can be seen to be trivially *not robust* when the hash function is given to the adversary. The adversary simply performs Gaussian elimination, discovering a “random collision”  $(x, y)$  in the function, where, with high probability  $x \oplus y$  will have large Hamming weight. This already breaks equation (1).

The situation is somewhat worse. Even in a very weak, oracle sense, corresponding to our evaluation-oracle-robustness definition, a result of Hardt and Woodruff [12] shows that there are no *linear functions*  $h$  that are robust for the gap- $\ell_2$  predicate. While their result does not carry over *as-is* to the setting of  $\ell_0$  (Hamming), we conjecture it does, leaving us with two options: (a) make the domain sparse: both the Gaussian elimination attack and the Hardt-Woodruff attack use the fact that Gaussian elimination is easy on the domain of the hash function; however making the domain sparse (say, the set of all strings of weight at most  $\beta n$  for some constant  $\beta < 1$ ) already rules it out; and (b) make the hash function non-linear: again, both attacks crucially exploit linearity. We will pursue both options, and as we will see, they are related.

But before we get there, let us ask whether we even need computational assumptions to get such a PPH. Can there be information-theoretic constructions? The first observation is that by a packing argument, if the output domain of the hash function has size less than  $2^{n - n \cdot H(\frac{d(1+\epsilon)}{n})} \approx 2^{n - d \log n(1+\epsilon)}$  (for small  $d$ ), there are bound to be “collisions”, namely, two far points (at distance more than  $d(1 + \epsilon)$ ) that hash to the same point. So, you really cannot compress much information-theoretically, especially as  $d$  becomes smaller. A similar bound holds when restricting the domain to strings of Hamming weight at most  $\beta n$  for constant  $\beta < 1$ .

With that bit of information, let us proceed to describe in a very high level our construction and the computational assumption. Our construction follows the line of thinking of Applebaum, Haramaty, Ishai, Kushilevitz and Vaikuntanathan [2] where they used the hardness of syndrome decoding problems to construct collision-resistant hash functions. Indeed, in a single sentence, our observation is that their collision-resistant hash functions are *locality-sensitive* by virtue of being *input-local*, and thus give us a robust gap-Hamming PPH (albeit under a different assumption).

In slightly more detail, our first step is to simply take the construction of Kushilevitz, Ostrovsky, and Rabani [19], and restrict the domain of the function. We show that finding two close points that get mapped to far points under the hash function is simply impossible (for our setting of parameters). On the other hand, there exist two far points that get mapped to close points under the hash functions (in fact, they even collide). Thus, showing that it is hard to find such points requires a computational assumption.

In a nutshell, our assumption says that given a random matrix  $\mathbf{A}$  where each entry is chosen from the Bernoulli distribution with  $\text{Ber}(1/d)$  with parameter  $1/d$ , it is hard to find a large Hamming weight vector  $\mathbf{x}$  where  $\mathbf{Ax} \pmod{2}$  has small Hamming weight. Of course, “large” and “small” here have to be parameterized correctly (see the full version for more details), however we observe that this is a generalization of the syndrome decoding assumption for low-density parity check (LDPC) codes, made by [2].

In our second step, we remove the sparsity requirement on the input domain of the predicate. We show a sparsification transformation which takes arbitrary  $n$ -bit vectors and outputs  $n' > n$ -bit *sparse* vectors such that (a) the transformation is injective, and (b) the

expansion introduced here does not cancel out the effect of compression achieved by the linear transformation  $\mathbf{x} \rightarrow \mathbf{A}\mathbf{x}$ . This requires careful tuning of parameters for which we refer the reader to the full version of this paper. Informally, our theorem statement is

► **Theorem 2** (Shortest-Vector PPH informal). *Let  $\lambda$  be a security parameter. Assuming the shortest-vector assumption (discussed above) with reasonable parameter settings, for any polynomial  $n = n(\lambda)$ , and any constants  $\epsilon, \eta > 0$ , there is an  $\eta$ -compressing robust property preserving hash family for GAPHAMMING( $n, d, \epsilon$ ) where  $d \leq \frac{n}{2 \log n}((1 - \epsilon) + (1 + \epsilon))$ .*

Notice that we get a better parameter  $d$  than from our first construction. This is, of course, because we make a stronger assumption. For more details, we refer the reader to the full version of this paper.

### The Necessity of Cryptographic Assumptions

The goal of robust PPH is to compress beyond the information theoretic limits, to a regime where incorrect hash outputs exist but are hard to find. If robustness is required even when the hash function is given, this inherently necessitates cryptographic hardness assumptions. A natural question is whether weaker forms of robustness (where the adversary sees only oracle access to the hash function) similarly require cryptographic assumptions, and what types of assumptions are required to build non-trivial PPHs of various kinds.

As a final contribution, we identify necessary assumptions for PPH for a kind of predicate we call *collision sensitive*. In particular, PPH for any such predicate in the double-oracle model implies the existence of one-way functions, and in the direct-access model implies existence of collision-resistant hash functions. In a nutshell, collision-sensitive means that finding a collision in the predicate breaks the property-preserving nature of any hash. The proof uses and expands on techniques from the work of Naor and Yaguev on adversarially robust Bloom Filters [24]. The basic idea is the same: without OWFs, we can invert arbitrary polynomially-computable functions with high probability in polynomial time, and using this we get a representation of the hash function/set, which can be used to find offending inputs.

## 2 Defining Property-Preserving Hash Functions

Our definition of property-preserving hash functions (PPHs) comes in several flavors, depending on whether we support total or partial predicates; whether the predicates take a single input or multiple inputs; and depending on the information available to the adversary. We discuss each of these choices in turn.

### Total vs. Partial Predicates

We consider *total predicates* that assign a 0 or 1 output to each element in the domain, and *promise (or partial) predicates* that assign a 0 or 1 to a subset of the domain and a wildcard (don't-care) symbol  $\star$  to the rest. More formally, a *total predicate*  $P$  on a domain  $X$  is a function  $P : X \rightarrow \{0, 1\}$ , well-defined as 0 or 1 for every input  $x \in X$ . A *promise predicate*  $P$  on a domain  $X$  is a function  $P : X \rightarrow \{0, 1, \star\}$ . Promise predicates can be used to describe scenarios (such as gap problems) where we only care about providing an exact answer on a subset of the domain.

Our definitions below will deal with the more general case of promise predicates, but we will discuss the distinction between the two notions when warranted.

### Single-Input vs Multi-Input Predicates

In the case of single-input predicates, we consider a class of properties  $\mathcal{P}$  and hash a single input  $x$  into  $h(x)$  in a way that given  $h(x)$ , one can compute  $P(x)$  for any  $P \in \mathcal{P}$ . Here,  $h$  is a compressing function. In the multi-input setting, we think of a single fixed property  $P$  that acts on a tuple of inputs, and require that given  $h(x_1), h(x_2), \dots, h(x_k)$ , one can compute  $P(x_1, x_2, \dots, x_k)$ . The second syntax is more expressive than the first, and so we use the multi-input syntax for constructions and the single-input syntax for lower bounds<sup>5</sup>.

Before we proceed to discuss robustness, we provide a working definition for a property-preserving hash function for the single-input syntax. For the multi-input predicate definition and further discussion, see the full version.

► **Definition 3.** A (non-robust)  $\eta$ -compressing Property Preserving Hash ( $\eta$ -PPH) family  $\mathcal{H} = \{h : X \rightarrow Y\}$  for a function  $\eta$  and a class of predicates  $\mathcal{P}$  requires the following two efficiently computable algorithms:

- $\mathcal{H}.\text{Samp}(1^\lambda) \rightarrow h$  is a randomized p.p.t. algorithm that samples a random hash function from  $\mathcal{H}$  with security parameter  $\lambda$ .
- $\mathcal{H}.\text{Eval}(h, P, y)$  is a deterministic polynomial-time algorithm that on input the hash function  $h$ , a predicate  $P \in \mathcal{P}$  and  $y \in Y$  (presumably  $h(x)$  for some  $x \in X$ ), outputs a single bit.

Additionally,  $\mathcal{H}$  must satisfy the following two properties:

- *$\eta$ -compressing*, namely,  $\log |Y| \leq \eta(\log |X|)$ , and
- *robust*, according to one of four definitions that we describe below, leading to four notions of PPH: definition 4 (non-robust PPH), 5 (evaluation-oracle-robust PPH or EO-PPH), 7 (double-oracle-robust PPH or DO-PPH), or 9 (direct-access robust PPH or DA-PPH). We will refer to the strongest form, namely direct-access robust PPH as simply robust PPH when the intent is clear from the context. See also Table 1 for a direct comparison between these.

### The Many Types of Robustness

We will next describe four definitions of robustness for PPHs, starting from the weakest to the strongest. Each of these definitions, when plugged into the last bullet of Definition 3, gives rise to a different type of property-preserving hash function. In each of these definitions, we will describe an adversary whose goal is to produce an input and a predicate such that the hashed predicate evaluation disagrees with the truth. The difference between the definitions is in what an adversary has access to, summarized in Table 1.

#### 2.1 Non-Robust PPH

We will start by defining the weakest notion of robustness which we call non-robust PPH. Here, the adversary has no information at all on the hash function  $h$ , and is required to produce a predicate  $P$  and a valid input  $x$ , namely where  $P(x) \neq \star$ , such that  $\mathcal{H}.\text{Eval}(h, P, x) \neq P(x)$  with non-negligible probability. When  $\mathcal{P}$  is the family of point functions (or equality functions), this coincides with the notion of 2-universal hash families [5]<sup>6</sup>.

<sup>5</sup> There is yet a third possibility, namely where there is a *fixed* predicate  $P$  that acts on a single input  $x$ , and we require that given  $h(x)$ , one can compute  $P(x)$ . This makes sense when the computational complexity of  $h$  is considerably less than that of  $P$ , say when  $P$  is the parity function and  $h$  is an  $AC^0$  circuit, as in the work of Dubrov and Ishai [10]. We do not explore this third syntax further in this work.

<sup>6</sup> While 2-universal hashing corresponds with a two-input predicate testing equality, the single-input version ( $\{P_{x_1}\}$  where  $P_{x_1}(x_2) = (x_1 == x_2)$ ) is more general, and so it is what we focus on.

## 16:10 Adversarially Robust Property-Preserving Hash Functions

■ **Table 1** A table comparing the adversary’s access to the hash function within different robustness levels of PPHs.

For security parameter  $\lambda$ , fixed predicate class  $\mathcal{P}$ , and  $h$  sampled from  $\mathcal{H}.\text{Samp}$

Non-Robust PPH	Adversary has no access to hash function or evaluation.
Evaluation-Oracle PPH	Access to the evaluation oracle $\mathcal{O}_h^{\text{Eval}}(x, P) = \mathcal{H}.\text{Eval}(h, P, h(x))$ .
Double-Oracle PPH	Access to both $\mathcal{O}_h^{\text{Eval}}$ (as above) and hash oracle $\mathcal{O}_h^{\text{Hash}}(x) = h(x)$ .
Robust PPH “Direct Access”	Direct access to the hash function, description of $h$ .

Here and in the following, we use the notation  $\Pr[A_1; \dots; A_m : E]$  to denote the probability that event  $E$  occurs following an experiment defined by executing the sequence  $A_1, \dots, A_m$  in order.

► **Definition 4.** A family of PPH functions  $\mathcal{H} = \{h : X \rightarrow Y\}$  for a class of predicates  $\mathcal{P}$  is a family of *non-robust* PPH functions if for any  $P \in \mathcal{P}$  and  $x \in X$  such that for  $P(x) \neq \star$ ,

$$\Pr[h \leftarrow \mathcal{H}.\text{Samp}(1^\lambda) : \mathcal{H}.\text{Eval}(h, P, h(x)) \neq P(x)] \leq \text{negl}(\lambda).$$

### 2.2 Evaluation-Oracle Robust PPH

In this model, the adversary has slightly more power than in the non-robust setting. Namely, she can adaptively query an oracle that has  $h \leftarrow \mathcal{H}.\text{Samp}(1^\lambda)$  in its head, on inputs  $P \in \mathcal{P}$  and  $x \in X$ , and obtain as output the hashed evaluation result  $\mathcal{H}.\text{Eval}(h, P, h(x))$ . Let  $\mathcal{O}_h(x, P) = \mathcal{H}.\text{Eval}(h, P, h(x))$ .

► **Definition 5.** A family of PPH functions  $\mathcal{H} = \{h : X \rightarrow Y\}$  for a class of predicates  $\mathcal{P}$  is a family of *evaluation-oracle robust (EO-robust)* PPH functions if, for any PPT adversary  $\mathcal{A}$ ,

$$\Pr[h \leftarrow \mathcal{H}.\text{Samp}(1^\lambda); (x, P) \leftarrow \mathcal{A}^{\mathcal{O}_h}(1^\lambda) : P(x) \neq \star \wedge \mathcal{H}.\text{Eval}(h, P, h(x)) \neq P(x)] \leq \text{negl}(\lambda).$$

The reader might wonder if this definition is very weak, and may ask if it follows just from the definition of a non-robust PPH family. In fact, for *total predicates*, we show that the two definitions are the same. At a high level, simply querying the evaluation oracle on (even adaptively chosen) inputs cannot reveal information about the hash function since with all but negligible probability, the answer from the oracle will be correct and thus simulatable without oracle access. The proof of the following lemma is in the full version.

► **Lemma 6.** *Let  $\mathcal{P}$  be a class of total predicates on  $X$ . A non-robust PPH  $\mathcal{H}$  for  $\mathcal{P}$  is also an Evaluation-Oracle robust PPH for  $\mathcal{P}$  for the same domain  $X$  and same codomain  $Y$ .*

However, when dealing with *promise* predicates, an EO-robustness adversary has the ability to make queries that do not satisfy the promise, and could get information about the hash function, perhaps even reverse-engineering the entire hash function itself. Indeed, Hardt and Woodruff [12] show that there are no EO-robust *linear* hash functions for a certain promise- $\ell_p$  distance property; whereas, non-robust linear hash functions for these properties follow from the work of Indyk [14, 13].



## 2.3 Double-Oracle PPH

We continue our line of thought, giving the adversary more power. Namely, she has access to two oracles, both have a hash function  $h \leftarrow \mathcal{H}.\text{Samp}(1^\lambda)$  in their head. The hash oracle  $\mathcal{O}_h^{\text{Hash}}$ , parameterized by  $h \in \mathcal{H}$ , outputs  $h(x)$  on input  $x \in X$ . The predicate evaluation oracle  $\mathcal{O}_h^{\text{Eval}}$ , also parameterized by  $h \in \mathcal{H}$ , takes as input  $P \in \mathcal{P}$  and  $y \in Y$  and outputs  $\mathcal{H}.\text{Eval}(h, P, y)$ . When  $\mathcal{P}$  is the family of point functions (or equality functions), this coincides with the notion of pseudo-random functions.

► **Definition 7.** A family of PPH functions  $\mathcal{H} = \{h : X \rightarrow Y\}$  for a class of predicates  $\mathcal{P}$  is a family of *double-oracle-robust PPH (DO-PPH)* functions if, for any PPT adversary  $\mathcal{A}$ ,

$$\Pr[h \leftarrow \mathcal{H}.\text{Samp}(1^\lambda); (x, P) \leftarrow \mathcal{A}^{\mathcal{O}_h^{\text{Hash}}, \mathcal{O}_h^{\text{Eval}}}(1^\lambda) : P(x) \neq \star \wedge \mathcal{H}.\text{Eval}(h, P, h(x)) \neq P(x)] \leq \text{negl}(\lambda).$$

We show that any evaluation-oracle-robust PPH can be converted into a double-oracle-robust PPH at the cost of a computational assumption, namely, one-way functions. In a nutshell, the observation is that the output of the hash function can be encrypted using a symmetric key that is stored as part of the hash description, and the evaluation proceeds by first decrypting. The proof of the following lemma is in the full version.

► **Lemma 8.** *Let  $\mathcal{P}$  be a class of (total or partial) predicates on  $X$ . Assume that one-way functions exist. Then, any EO-robust PPH for  $\mathcal{P}$  can be converted into a DO-robust PPH for  $\mathcal{P}$ .*

## 2.4 Direct-Access Robust PPH

Finally, we define the strongest notion of robustness where the adversary is given the description of the hash function itself. When  $\mathcal{P}$  is the family of point functions (or equality functions), this coincides with the notion of collision-resistant hash families.

► **Definition 9.** A family of PPH functions  $\mathcal{H} = \{h : X \rightarrow Y\}$  for a class of predicates  $\mathcal{P}$  is a family of *direct-access robust PPH* functions if, for any PPT adversary  $\mathcal{A}$ ,

$$\Pr[h \leftarrow \mathcal{H}.\text{Samp}(1^\lambda); (x, P) \leftarrow \mathcal{A}(h) : P(x) \neq \star \wedge \mathcal{H}.\text{Eval}(h, P, h(x)) \neq P(x)] \leq \text{negl}(\lambda).$$

We will henceforth focus on *direct-access-robust* property-preserving hash functions and refer to them simply as robust PPHs.

## 3 Property-Preserving Hashing and Communication Complexity

In this section, we identify and examine a relationship between property-preserving hash families (in the single-input syntax) and protocols in the one-way communication (OWC) model. A OWC protocol is a protocol between two players, Alice and Bob, with the goal of evaluating a certain predicate on their inputs and with the restriction that only Alice can send messages to Bob.

Our first observation is that non-robust property-preserving hash functions and OWC protocols [31] are equivalent except for two changes. First, PPHs require the parties to be computationally efficient, and second, PPHs also require protocols that incur error negligible in a security parameter. It is also worth noting that while we can reference lower-bounds in

the OWC setting, these lower bounds are typically of the form  $\Omega(n)$  and are not exact. On the other hand, in the PPH setting, we are happy with getting a single bit of compression, and so an  $\Omega(n)$  lower bound still does not tell us whether or not a PPH is possible. So, while we can use previously known lower bounds for some well-studied OWC predicates, we need to refine them to be exactly  $n$  in the presence of negligible error. We also propose a framework (for total predicates) that yields exactly  $n$  lower bounds for `INDEXn`, `GREATERTHAN`, and `EXACTHAMMING`.

### 3.1 PPH Lower Bounds from One-Way Communication Lower Bounds

In this section, we will review the definition of OWC, and show how OWC lower bounds imply PPH impossibility results.

► **Definition 10.** [31, 18] A  $\delta$ -error public-coin OWC protocol  $\Pi$  for a two-input predicate  $P : \{0, 1\}^n \times \{0, 1\}^n \rightarrow \{0, 1\}$  consists of a space  $R$  of randomness, and two functions  $g_a : X_1 \times R \rightarrow Y$  and  $g_b : Y \times X_2 \times R \rightarrow \{0, 1\}$  so that for all  $x_1 \in X_1$  and  $x_2 \in X_2$ ,

$$\Pr[r \leftarrow R; y = g_a(x_1; r) : g_b(y, x_2; r) \neq P(x_1, x_2)] \leq \delta.$$

A  $\delta$ -error public-coin OWC protocol  $\Pi$  for a *class of predicates*  $\mathcal{P} = \{P : \{0, 1\}^n \rightarrow \{0, 1\}\}$ , is defined much the same as above, with a function  $g_a : X \times R \rightarrow Y$ , and another function  $g_b : Y \times \mathcal{P} \times R \rightarrow \{0, 1\}$ , which instead of taking a second input, takes a predicate from the predicate class. We say  $\Pi$  has  $\delta$ -error if

$$\Pr[r \leftarrow R; y = g_a(x; r) : g_b(y, P; r) \neq P(x)] \leq \delta$$

Let  $\text{Protocols}_\delta(P)$  denote the set of OWC protocols with error at most  $\delta$  for a predicate  $P$ , and for every  $\Pi \in \text{Protocols}_\delta(P)$ , let  $Y_\Pi$  be the range of messages Alice sends to Bob (the range of  $g_a$ ) for protocol  $\Pi$ .

► **Definition 11.** The randomized, public-coin *OWC complexity* of a predicate  $P$  with error  $\delta$ , denoted  $R_\delta^{A \rightarrow B}(P)$ , is the minimum over all  $\Pi \in \text{Protocols}_\delta(P)$  of  $\lceil \log |Y_\Pi| \rceil$ .

For a predicate class  $\mathcal{P}$ , we define the randomized, public-coin OWC complexity with error  $\delta$ , denoted  $R_\delta^{A \rightarrow B}(\mathcal{P})$ , is the minimum over all  $\Pi \in \text{Protocols}_\delta(\mathcal{P})$  of  $\lceil \log |Y_\Pi| \rceil$ .

A PPH scheme for a two-input predicate<sup>7</sup>  $P$  yields a OWC protocol for  $P$  with communication comparable to a single hash output size.

► **Theorem 12.** *Let  $P$  be any two-input predicate  $P$  and  $\mathcal{P} = \{P_x\}_{x \in \{0, 1\}^n}$  be the corresponding predicate class where  $P_{x_2}(x_1) = P(x_1, x_2)$ . Now, let  $\mathcal{H}$  be a PPH in any model for  $\mathcal{P}$  that compresses  $n$  bits to  $m = \eta n$ . Then, there exists a OWC protocol  $\Pi$  such that the communication of  $\Pi$  is  $m$  and with negligible error.*

*Conversely, the amount of possible compression of any (robust or not) PPH family  $\mathcal{H} : \{h : X \rightarrow Y\}$  is lower bounded by  $R_{\text{negl}(\lambda)}^{A \rightarrow B}(P)$ . Namely,  $\log |Y| \geq R_{\text{negl}(\lambda)}^{A \rightarrow B}(\mathcal{P})$ .*

Essentially, the OWC protocol is obtained by using the public common randomness  $r$  to sample a hash function  $h = \mathcal{H}.\text{Samp}(1^\lambda; r)$ , and then Alice simply sends the hash  $h(x_1)$  of her input to Bob. (Proof in full version.)

<sup>7</sup> Or rather, for the induced class of single-input predicates  $\mathcal{P} = \{P_{x_2}\}_{x_2 \in \{0, 1\}^n}$ , where  $P_{x_2}(x_1) = P(x_1, x_2)$ ; we will use these terminologies interchangeably.



### 3.2 OWC and PPH lower bounds for Reconstructing Predicates

We next leverage this connection together with OWC lower bounds to obtain impossibility results for PPHs. First, we will discuss the total predicate case; we consider some partial predicates in section 3.3.

As discussed, to demonstrate the impossibility of a PPH, one must give an explicit  $n$ -bit communication complexity lower bound (not just  $\Omega(n)$ ) for negligible error. We give such lower bounds for an assortment of predicate classes by a general approach framework we refer to as *reconstructing*. Intuitively, a predicate class is reconstructing if, when given only access to predicates evaluated on an input  $x$ , one can, in polynomial time, determine the exact value of  $x$  with all but negligible probability.

► **Definition 13.** A class  $\mathcal{P}$  of total predicates  $P : \{0, 1\}^n \rightarrow \{0, 1\}$ , is *reconstructing* if there exists a PPT algorithm  $L$  (a ‘learner’) such that for all  $x \in \{0, 1\}^n$ , given randomness  $r$  and oracle access to predicates  $\mathcal{P}$  on  $x$ , denoted  $\mathcal{O}_x(P) = P(x)$ ,

$$\Pr_r[L^{\mathcal{O}_x}(r) \rightarrow x] \geq 1 - \text{negl}(n).$$

► **Theorem 14.** If  $\mathcal{P}$  is a reconstructing class of predicates on input space  $\{0, 1\}^n$ , then a PPH does not exist for  $\mathcal{P}$ .

The full proof appears in the full version. The main idea is to simulate the learner  $L$  on two different inputs  $x_1$  and  $x_2$ : at some query,  $L$  must get a different answer from the oracle, differentiating  $x_1$  from  $x_2$ . Since  $h$  is compressing, there are many  $x_1$  and  $x_2$  that collide on the same output. We show that we can guess such an  $x_1$  that is paired with an  $x_2$ , and the query that they differ on with  $1/\text{poly}(n)$  probability. Once we do that, the oracle must answer incorrectly for one of  $x_1$  or  $x_2$ , and there is a half chance that we chose the  $x_i$  that evaluated incorrectly.

#### Reconstructing using $\text{Index}_n$ , $\text{GreaterThan}$ , or $\text{ExactHamming}$

We turn to specific examples of predicate classes and sketch why they are reconstructing. For formal proofs, we refer the reader to the full version of this paper.

- The  $\text{INDEX}_n$  class of predicates  $\{P_1, \dots, P_n\}$  is defined over  $x \in \{0, 1\}^n$  where  $P_i(x) = x_i$ , the  $i$ ’th bit of  $x$ .  $\text{INDEX}_n$  is reconstructing simply because the learner  $L$  can just query the each of the  $n$  indices of the input and exactly reconstruct:  $x_i = P_i(x)$ .
- The  $\text{GREATERTHAN}$  class of predicates  $\{P_x\}_{x \in [2^n]}$  is defined over  $x \in [2^n] = \{0, 1\}^n$  where  $P_{x_2}(x_1) = 1$  if  $x_1 > x_2$  and 0 otherwise.  $\text{GREATERTHAN}$  is reconstructing because we can run a binary search on the input space, determining the exact value of  $x$  in  $n$  queries.  $\text{GREATERTHAN}$  is an excellent example for how an adaptive learner  $L$  can reconstruct.
- The  $\text{EXACTHAMMING}(\alpha)$  class of predicates  $\{P_x\}_{x \in \{0, 1\}^n}$  is defined over  $x \in \{0, 1\}^n$  where  $P_{x_2}(x_1) = 1$  if  $\|x_1 - x_2\|_0 > \alpha$  and 0 otherwise. To show that  $\text{EXACTHAMMING}(n/2)$  is reconstructing requires a little more work. The learner  $L$  resolves each index of  $x$  independently. For each index,  $L$  makes polynomially many random-string queries  $\mathbf{r}$  to  $\mathcal{O}_x$ ; if the  $i$ ’th bit of  $\mathbf{r}$  equals  $x_i$ , then  $\mathbf{r}$  is more likely to be within  $n/2$  hamming distance of  $\mathbf{x}$ , and if the bits are different,  $\mathbf{r}$  is more likely to not be within  $n/2$  hamming distance of  $\mathbf{x}$ . The proof uses techniques from [15], and is an example where the learner uses randomness to reconstruct.

We note that it was already known that  $\text{INDEX}_n$  and  $\text{EXACTHAMMING}(n/2)$  had OWC complexity of  $n$ -bits for any negligible error [18], though no precise lower bound for randomized OWC protocols was known for  $\text{GREATERTHAN}$ . What is new here is our unified framework.

### 3.3 Lower bounds for some partial predicates

In the previous section, we showed how the ability to reconstruct an input using a class of total predicates implied that PPHs for the class cannot exist. This general framework, unfortunately, does not directly extend to the partial-predicate setting, since it is unclear how to define the behavior of an oracle for the predicate. Nevertheless, we can still take existing OWC lower bounds and their techniques to prove impossibility results in this case. We will show that  $\text{GAPHAMMING}(n, n/2, 1/\sqrt{n})$  (the promise version of  $\text{EXACTHAMMING}$ ) cannot admit a PPH, and that while  $\text{Gap-}k$   $\text{GREATERTHAN}$  has a perfectly correct PPH compressing to  $n - \log(k) - 1$  bits, compressing any further results in polynomial error (and thus no PPH with more compression).

First, we define these partial predicates.

- **Definition 15.** The definitions for  $\text{GAPHAMMING}(n, d, \epsilon)$  and  $\text{Gap-}k$   $\text{GREATERTHAN}$  are:
  - The  $\text{GAPHAMMING}(n, d, \epsilon)$  class of predicates  $\{P_x\}_{x \in \{0,1\}^n}$  has  $P_{x_2}(x_1) = 1$  if  $\|x_1 - x_2\|_0 \geq d(1 + \epsilon)$ , 0 if  $\|x_1 - x_2\|_0 \leq d(1 - \epsilon)$ , and  $\star$  otherwise.
  - The  $\text{Gap-}k$   $\text{GREATERTHAN}$  class of predicates  $\{P_x\}_{x \in [2^n]}$  has  $P_{x_2}(x_1) = 1$  if  $x_1 > x_2 + k$ , 0 if  $x_1 < x_2 - k$ , and  $\star$  otherwise.

Now, we provide some intuition for why these lower bounds (and the upper bound) exist.

#### Gap-Hamming

Our lower bound will correspond to a refined OWC lower bound for the Gap-Hamming problem in the relevant parameter regime. Because we want to prove that we cannot even compress by a single bit, we need to be careful with our reduction: we want the specific parameters for which we have a lower bound, and must keep close track of how the error changes within our reduction.

- **Theorem 16.** *There does not exist a PPH for  $\text{GAPHAMMING}(n, n/2, 1/\sqrt{n})$ .*

To prove, we show the OWC complexity  $R_{\text{negl}(n)}^{A \rightarrow B}(\text{GAPHAMMING}(n, n/2, 1/\sqrt{n})) = n$ . A  $\Omega(n)$  OWC lower bound for Gap-Hamming in this regime has been proved in a few different ways [29, 30, 15]. Our proof will be a refinement of [15] and is detailed in the full version.

The high-level structure of the proof is to reduce  $\text{INDEX}_n$  to  $\text{GAPHAMMING}$  with the correct parameters. Very roughly, the  $i$ th coordinate of an input  $x \in \{0, 1\}^n$  can be inferred from the bias it induces on the Hamming distance between  $x$  and random public vectors. The reduction adds negligible error, but since we require  $n$  bits for negligible-error  $\text{INDEX}_n$ , we also require  $n$  bits for a OWC protocol for  $\text{GAPHAMMING}$ .

Notice that this style of proof looks morally as though we are “reconstructing” the input  $x$  using  $\text{INDEX}_n$ . However, the notion of getting a reduction from  $\text{INDEX}_n$  to another predicate-class in the OWC model is not the same as being able to query an oracle about the predicate and reconstruct based off of oracle queries. Being able to make a similar reconstructing characterization of partial-predicates as we have for total predicates would be useful and interesting in proving further lower bounds.

### Gap- $k$ GreaterThan

This predicate is a natural extension of GREATER THAN: we only care about learning that  $x_1 < x_2$  if  $|x_1 - x_2|$  is larger than  $k$  (the gap). Intuitively, a hash function can maintain this information by simply removing the  $\log(k)$  least significant bits from inputs and directly comparing: if  $h(x_1) = h(x_2)$ , they can be at most  $k$  apart. We can further remove one additional bit using the fact that we know  $x_2$  when given  $h(x_1)$  (considering Gap- $k$  GreaterThan as the corresponding predicate class parameterized by  $x_2$ ).

For the lower bound, we prove a OWC lower bound, showing  $R_{\text{negl}(n)}^{A \rightarrow B}(\mathcal{P}) = n - \log(k) - 1$ . This will be a proof by contradiction: if we compress to  $n - \log(k) - 2$  bits, we obtain many collisions that are more than  $3.5k$  apart. These far collisions imply the existence of inputs that the OWC protocol must fail on, even given the gap. We are able to find these inputs the OWC must fail on with polynomial probability, and this breaks the all-but-negligible correctness of the protocol. Our formal theorem statement is below.

► **Theorem 17.** *There exists a PPH with perfect correctness for Gap- $k$  GREATER THAN compressing from  $n$  bits to  $n - \log(k) - 1$ . This is tight: no PPH for Gap- $k$  GREATER THAN can compress to fewer than  $n - \log(k) - 1$  bits.*

For the proof, see the full version. To understand the construction of the PPH, first consider a hash function that just chops off the least-significant  $\log(k)$  bits from the inputs. Clearly, comparing two hashed values for which is greater than the other gives the gap- $k$ -GREATER THAN result correctly. In order to get the last bit off, we chop off the least-significant  $\log(k)$  bits and then round up or down depending on the  $\log(k) + 1$ 'th significant bit before removing it. Evaluation is almost essentially comparing hashes, but we exploit the fact that we know the entire second input ( $x_2$ ), which is why we are able to remove one extra bit. Proving the lower bound is a bit trickier, but essentially involves simulating a binary search.

## 4 A Gap-Hamming PPH from Collision Resistance

In this section, we present one of our constructions for a PPH for the GAPHAMMING problem. Recall from section 3.3 that the gap-Hamming property  $P = \text{GAPHAMMING}(n, d, \epsilon)$  is parameterized by the input domain  $\{0, 1\}^n$ , an integer  $d \in [n]$  and a parameter  $\epsilon \in \mathbb{R}^{\geq 0}$ , so that  $P(\mathbf{x}_1, \mathbf{x}_2) = 1$  if  $\|\mathbf{x}_1 \oplus \mathbf{x}_2\|_0 \geq d(1 + \epsilon)$  and 0 if  $\|\mathbf{x}_1 \oplus \mathbf{x}_2\|_0 \leq d(1 - \epsilon)$ . Both of our constructions will distinguish between  $d(1 - \epsilon)$ -close and  $d(1 + \epsilon)$ -far vectors for  $d \approx O(n/\log n)$ . This means that the gap is quite large, approximately  $O(n/\log n)$ .

This construction is a robust  $m/n$ -compressing  $\text{GAPHAMMING}(n, d, \epsilon)$  PPH for any  $m = n^{\Omega(1)}$ ,  $d = o(n/\log \lambda)$  and any constant  $\epsilon > 0$ . Security of the construction holds under the (standard) assumption that collision-resistant hash function families (CRHFs) exist.

We now informally describe the idea of the construction which, in one word, is “sub-sampling”. In slightly more detail, the intuition is to notice that if  $\mathbf{x}_1 \in \{0, 1\}^n$  and  $\mathbf{x}_2 \in \{0, 1\}^n$  are *close*, then *most small enough subsets* of indices of  $\mathbf{x}_1$  and  $\mathbf{x}_2$  will match identically. On the other hand, if  $\mathbf{x}_1$  and  $\mathbf{x}_2$  are *far*, then most *large enough subsets* of indices will differ. This leads us to the first idea for the construction, namely, fix a collection of sets  $\mathcal{S} = \{S_1, \dots, S_k\}$  where each  $S_i \subseteq [n]$  is a subset of appropriately chosen size  $s$ . On input  $\mathbf{x} \in \{0, 1\}^n$ , output  $\mathbf{y} = (\mathbf{x}|_{S_1}, \dots, \mathbf{x}|_{S_k})$  where  $\mathbf{x}|_S$  denotes the substring of  $\mathbf{x}$  indexed by the set  $S$ . The observation above tells us that if  $\mathbf{x}_1$  and  $\mathbf{x}_2$  are close (resp. far), so are  $\mathbf{y}_1$  and  $\mathbf{y}_2$ .

However, this does not compress the vector  $\mathbf{x}$ . Since the union of all the sets  $\bigcup_{i \in [k]} S_i$  has to be the universe  $[n]$  (or else, finding a collision is easy), it turns out that we are just comparing the vectors index-by-index. Fortunately, it is not necessary to output  $\mathbf{x}|_{S_i}$  by

themselves; rather we can simply output the collision-resistant hashes. That is, we will let the PPH hash of  $\mathbf{x}$ , denoted  $\mathbf{y}$ , be  $(g(\mathbf{x}|_{S_1}), \dots, g(\mathbf{x}|_{S_k}))$  where  $g$  is a collision resistant hash function randomly drawn from a CRHF family.

This simple construction works as long as  $s$ , the size of the sets  $S_i$ , is  $\Theta(n/d)$ , and the collection  $\mathcal{S}$  satisfies that any subset of disagreeing input indices  $T \subseteq [n]$  has nonempty intersection with roughly the corresponding fraction of subsets  $S_i$ . The latter can be achieved by selecting the  $S_i$  of size  $\Theta(n/d)$  at random, or alternatively as defined by the neighbor sets of a bipartite expander. We are additionally constrained by the fact that the CRHF must be secure against adversaries running in time  $\text{poly}(\lambda)$ . So, let  $t = t(\lambda)$  be the smallest output size of the CRHF such that it is secure against  $\text{poly}(\lambda)$ -time adversaries. Since the input size  $s$  to the CRHF must be  $\omega(t)$  so that  $g$  actually compresses, this forces  $n/d = \omega(t)$ , and thus  $d = o(n/t)$ .

Before presenting our construction more formally, we define our tools.

- We will use a family of CRHFs that take inputs of variable size and produce outputs of  $t$  bits and denote it by  $\mathcal{H}_t = \{h : \{0, 1\}^* \rightarrow \{0, 1\}^t\}$ . We implicitly assume a procedure for sampling a seed for the CRHF given a security parameter  $1^\lambda$ . One could set  $t = \omega(\log \lambda)$  and assume the exponential hardness of the CRHF, or set  $t = \lambda^{O(1)}$  and assume polynomial hardness. These choices will result in different parameters of the PPH hash function.
- We will use an  $(n, k, D, \gamma, \alpha)$ -bipartite expander  $G = (L \cup R, E)$  which is a  $D$ -left-regular bipartite graph, with  $|L| = n$  and  $|R| = k$  such that for every  $S \subset L$  for which  $|S| \leq \gamma n$ , we have  $|N(S)| \geq \alpha|S|$ , where  $N(S)$  is the set of neighbors of  $S$ . For technical reasons, we will need the expander to be  $\delta$ -balanced on the right, meaning that for every  $v \in R$ ,  $|N(v)| \geq (1 - \delta)nD/k$ .

A simple probabilistic construction shows that for every  $n \in \mathbb{N}$ ,  $k = o(n)$  and constant  $a \in (0, 1)$ , and any  $\gamma = o(\frac{k}{n \log(n/k)})$  and  $D = \Theta(\log(1/\gamma))$  so that for every  $\delta > 0$ ,  $\delta$ -balanced  $(n, k, D, \gamma, \alpha)$ -bipartite expanders exist. In fact, there are even explicit efficient constructions that match these parameters [4].

The construction is in Table 2. We first discuss explicit parameter settings.

### Setting the Parameters

The parameters required for this construction to be secure and constructible are as follows.

- Let  $n \in \mathbb{N}$  and constant  $\epsilon > 0$ .
- We require two building blocks: a CRHF and an expander. So, let  $\mathcal{H}_t = \{g : \{0, 1\}^* \rightarrow \{0, 1\}^t\}$  be a family of CRHFs secure against  $\text{poly}(\lambda)$ -time adversaries. Let  $G$  be a  $\delta$ -balanced  $(n, k, D, \gamma, \alpha)$ -expander for a constant  $\delta$  bounding the degree of the right-nodes, where  $n$  is the size of the left side of the graph,  $k$  is the size of the right,  $D$  is the left-degree,  $\gamma n$  is the upper bound for an expanding set on the right that expands to a set of size  $\alpha$  times the original size.
- These building blocks yield the following parameters for the construction: compression is  $\eta = kt/n$  and our center for the Gap-Hamming problem is bounded by  $\frac{\gamma n}{(1+\epsilon)} \leq d < \frac{k}{D(1-\epsilon)}$ .

If we consider what parameter settings yield secure CRHFs with output size  $t$  and for what parameters we have expanders, we have a PPH construction where given any  $n$  and  $\epsilon$ , there exists a  $d = o(n)$  and  $\eta = O(1)$  such that Construction 2 is a robust PPH for gap-Hamming. We will see that the smaller  $t$  is, the stronger the CRHF security assumption, but the better our compression.

Now, given these settings of parameters, we have the following theorem that Construction 2 is a robust Gap-Hamming PPH.

■ **Table 2** Construction 2 of a robust  $\text{GAPHAMMING}(n, d, \epsilon)$  PPH family from CRHFs. Resulting parameters  $(n, m, d, \epsilon)$  discussed in text.

<u>Robust <math>\text{GAPHAMMING}(n, d, \epsilon)</math> PPH family <math>\mathcal{H}</math> from any CRHF</u>	
Our $(n, m, d, \epsilon)$ -robust PPH family $\mathcal{H} = (\mathcal{H}.\text{Samp}, \mathcal{H}.\text{Eval})$ is defined as follows.	
■ $\mathcal{H}.\text{Samp}(1^\lambda, n)$ . Fix a $\delta$ -balanced $(n, k, D, \gamma, \alpha)$ -bipartite expander $G = (L \cup R, E)$ (either deterministically or probabilistically). Sample a CRHF $g \leftarrow \mathcal{H}_t$ . Output $h = (G, g)$ .	
■ $\mathcal{H}.\text{Hash}(h = (G, g), \mathbf{x})$ . For every $i \in [k]$ , compute the (ordered) set of neighbors of the $i$ -th right vertex in $G$ , denoted $N(i)$ . Let $\hat{\mathbf{x}}^{(i)} := \mathbf{x} _{N(i)}$ be $\mathbf{x}$ restricted to the set $N(i)$ . Output	
$h(\mathbf{x}) := (g(\hat{\mathbf{x}}^{(1)}), \dots, g(\hat{\mathbf{x}}^{(k)}))$	
as the hash of $\mathbf{x}$ .	
■ $\mathcal{H}.\text{Eval}(h = (G, g), \mathbf{y}_1, \mathbf{y}_2)$ . Compute the threshold $\tau = D \cdot d \cdot (1 - \epsilon)$ . Parse $\mathbf{y}_1 = (\hat{\mathbf{y}}_1^{(1)}, \dots, \hat{\mathbf{y}}_1^{(k)})$ and $\mathbf{y}_2 = (\hat{\mathbf{y}}_2^{(1)}, \dots, \hat{\mathbf{y}}_2^{(k)})$ . Compute	
$\Delta' = \sum_{i=1}^k \text{Ind}(\hat{\mathbf{y}}_1^{(i)} \neq \hat{\mathbf{y}}_2^{(i)}),$	
where $\text{Ind}$ denotes the indicator predicate. If $\Delta' \leq \tau$ , output <b>CLOSE</b> . Otherwise, output <b>FAR</b> .	

► **Theorem 18.** *Let  $\lambda$  be a security parameter. Assuming that exponentially secure CRHFs exist, then for any polynomial  $n = n(\lambda)$ , and any constants  $\epsilon, \eta > 0$ , Construction 2 is an  $\eta$ -compressing robust property preserving hash family for  $\text{GAPHAMMING}(n, d, \epsilon)$  where  $d = o(n/\log \lambda \log \log \lambda)$ . Assuming only that polynomially secure CRHFs exist, for any constant  $c > 0$ , we achieve  $d = o(n/\lambda^c)$ .*

**Proof.** Before getting into the proof, we more explicitly define the parameters, including parameters associated with the expander in our construction:

1. Let  $n \in \mathbb{N}$  be the input size and let  $\epsilon > 0$  be any constant.
2. Our CRHF is  $\mathcal{H}_t = \{g : \{0, 1\}^* \rightarrow \{0, 1\}^t\}$ .
3. Our expander will be a  $\delta$ -balanced  $(n, k, D, \gamma, \alpha)$ -expander, where  $k < n/t$ ,  $\gamma = o(\frac{k}{n \log(n/k)})$ ,  $D = \Theta(\log(1/\gamma))$ , and  $\alpha > D \cdot \frac{d(1-\epsilon)}{\gamma n}$ .
4. Our center for the gap-hamming problem is  $d$ , and is constrained by  $\frac{\gamma n}{1+\epsilon} \leq d < \frac{k}{D(1-\epsilon)}$ .
5. Constraint 4 implies that  $k = n^{\Omega(1)}$ , since  $\frac{\gamma n \cdot D(1-\epsilon)}{1+\epsilon} < k$ .

Now, we prove our construction is well-defined and efficient. Fix any  $\delta, a \in (0, 1)$ . In the full version, we explicitly prove that  $\delta$ -balanced  $(n, k, D, \gamma, \alpha = an)$ -bipartite expanders exist and can be efficiently sampled for  $k = o(n/\log n)$ ,  $D = \Theta(\log \log n)$ , and  $\gamma = \tilde{\Theta}(1/\log n)$ . Thus, sampling the graph  $G$  before running the construction is efficient. Once we have a  $G$ , sampling and running a CRHF  $k = O(n)$  times is efficient. Comparing  $k$  outputs of the hash function is also efficient. Therefore, each of  $\mathcal{H}.\text{Samp}$ ,  $\mathcal{H}.\text{hash}$ , and  $\mathcal{H}.\text{Eval}$  is efficient in  $\lambda = \text{poly}(n)$ .

Now, we prove that Construction 2 is compressing. Points 2 and 3 mean that  $m = k \cdot t < n/t \cdot t = n$ , as required.

Lastly, we will prove our construction is robust. Let  $\mathcal{A}$  be a PPT adversary. We will show that  $\mathcal{A}$  (in fact, even an unbounded adversary) cannot find  $\mathbf{x}_1$  and  $\mathbf{x}_2$  such that  $\|\mathbf{x}_1 - \mathbf{x}_2\| \leq d(1 - \epsilon)$  but  $\mathcal{H}.\text{Eval}(h, h(\mathbf{x}_1), h(\mathbf{x}_2))$  evaluates to **FAR**, and that  $\mathcal{A}$  must break the collision-resistance of  $\mathcal{H}_t$  in order to find  $\mathbf{x}_1$  and  $\mathbf{x}_2$  where  $\|\mathbf{x}_1 - \mathbf{x}_2\| \geq d(1 + \epsilon)$  but  $\mathcal{H}.\text{Eval}(h, h(\mathbf{x}_1), h(\mathbf{x}_2))$  evaluates to **CLOSE**.

- First, consider any  $\mathbf{x}_1, \mathbf{x}_2 \in \{0, 1\}^n$  where  $\|\mathbf{x}_1 - \mathbf{x}_2\|_0 \leq d(1 - \epsilon)$ . Let  $\Delta = \|\mathbf{x}_1 - \mathbf{x}_2\|_0$ . So, consider the set  $S \subset L$  corresponding to the indices that are different between  $\mathbf{x}_1$  and  $\mathbf{x}_2$ , and  $T = N(S) \subset R$ . The maximum size of  $T$  is  $|S| \cdot D$ , the degree of the graph. For every  $i \in T$ , we get that the intermediate computation has  $\hat{\mathbf{x}}_1^{(i)} \neq \hat{\mathbf{x}}_2^{(i)}$ , but for every  $j \notin T$ , we have  $\hat{\mathbf{x}}_1^{(j)} = \hat{\mathbf{x}}_2^{(j)}$  which implies  $\hat{\mathbf{y}}_1^{(j)} = \hat{\mathbf{y}}_2^{(j)}$  after applying  $g$ . Therefore  $\sum_{i=1}^k \text{Ind}(\hat{\mathbf{y}}_1^{(i)} \neq \hat{\mathbf{y}}_2^{(i)}) \leq \sum_{i \in S} \text{Ind}(\hat{\mathbf{y}}_1^{(i)} \neq \hat{\mathbf{y}}_2^{(i)}) + \sum_{j \notin S} \text{Ind}(\hat{\mathbf{y}}_1^{(j)} \neq \hat{\mathbf{y}}_2^{(j)}) \leq \Delta \cdot D$ . We set the threshold  $\tau = D \cdot d \cdot (1 - \epsilon)$  in the evaluation. Point 4 guarantees that  $\tau < k$  (and implicitly implies  $k > D(1 - \epsilon)$ ), so because  $D \cdot \Delta \leq D \cdot d(1 - \epsilon) = \tau < k$ ,  $\mathcal{H}.\text{Eval}$  will evaluate  $\Delta' \leq \tau$ . Thus,  $\mathcal{H}.\text{Eval}$  will always evaluate to **CLOSE** in this case, regardless of the choice of CRHF.

- Now consider  $\|\mathbf{x}_1 - \mathbf{x}_2\|_0 \geq d(1 + \epsilon)$ , and again, let  $\Delta = \|\mathbf{x}_1 - \mathbf{x}_2\|_0$  and define  $S \subset L$  and  $T \subset R$  as before.

By point 4 again ( $\gamma n \leq d(1 + \epsilon)$ ), we can restrict  $S$  to  $S'$  where  $|S'| = \gamma n$ , and by the properties of expanders  $|N(S')| \geq \gamma n \cdot \alpha$ . Now, point 3 guarantees that  $\tau = D \cdot d \cdot (1 - \epsilon) < \alpha \cdot \gamma n$ . So, for every  $i \in T'$ ,  $\hat{\mathbf{x}}_1^{(i)} \neq \hat{\mathbf{x}}_2^{(i)}$ , and  $|T'| \geq \alpha \cdot \gamma n > \tau$ . Now we want to argue that with all but negligible probability over our choice of  $g$ ,  $g$  will preserve this equality relation, and so  $\Delta' = |T'|$ . Given that our expander is  $\delta$ -balanced for some constant  $\delta > 0$ , we have that  $|\hat{\mathbf{x}}_1^{(i)}| = |\hat{\mathbf{x}}_2^{(i)}| = |N(r_i)| \geq (1 - \delta)nD/k$ . Now, point 3 states that the constraints have  $k < n/t$ , implying  $n/k > t$ . So,  $(1 - \delta)D \cdot n/k > (1 - \delta)D \cdot t$ .

This means that every input to  $g$  will be larger than the output ( $(1 - \delta)$  is a constant and  $D = \omega(1)$ ), and so if  $g(\hat{\mathbf{x}}_1^{(i)}) = g(\hat{\mathbf{x}}_2^{(i)})$  but  $\hat{\mathbf{x}}_1^{(i)} \neq \hat{\mathbf{x}}_2^{(i)}$  for any  $i$ , then our adversary has found a collision, which happens with all but negligible probability for adversaries running in time  $\text{poly}(\lambda)$ .

Therefore, with all but negligible probability over the choice of  $g$  and adversarially chosen  $\mathbf{x}_1$  and  $\mathbf{x}_2$  in this case,  $\Delta' = \sum_{i=1}^{m'} \text{Ind}(\hat{\mathbf{y}}_1^{(i)} \neq \hat{\mathbf{y}}_2^{(i)}) \geq \alpha \cdot \gamma n = \tau$ , and  $\mathcal{H}.\text{Eval}$  outputs **FAR**. ◀

---

## References

- 1 Noga Alon, Yossi Matias, and Mario Szegedy. The Space Complexity of Approximating the Frequency Moments. In *Proceedings of the Twenty-Eighth Annual ACM Symposium on the Theory of Computing, Philadelphia, Pennsylvania, USA, May 22-24, 1996*, pages 20–29, 1996.
- 2 Benny Applebaum, Naama Haramaty, Yuval Ishai, Eyal Kushilevitz, and Vinod Vaikuntanathan. Low-Complexity Cryptographic Hash Functions. In *8th Innovations in Theoretical Computer Science Conference, ITCS 2017, January 9-11, 2017, Berkeley, CA, USA*, pages 7:1–7:31, 2017.
- 3 László Babai, Anna Gál, Peter G. Kimmel, and Satyanarayana V. Lokam. Communication Complexity of Simultaneous Messages. *SIAM J. Comput.*, 33(1):137–166, 2003.
- 4 Michael Capalbo, Omer Reingold, Salil Vadhan, and Avi Wigderson. Randomness Conductors and Constant-degree Lossless Expanders. In *Proceedings of the Thirty-fourth Annual ACM Symposium on Theory of Computing, STOC '02*, pages 659–668, New York, NY, USA, 2002. ACM. doi:10.1145/509907.510003.



- 5 Larry Carter and Mark N. Wegman. Universal Classes of Hash Functions (Extended Abstract). In *Proceedings of the 9th Annual ACM Symposium on Theory of Computing, May 4-6, 1977, Boulder, Colorado, USA*, pages 106–112, 1977.
- 6 Moses Charikar, Kevin C. Chen, and Martin Farach-Colton. Finding frequent items in data streams. *Theor. Comput. Sci.*, 312(1):3–15, 2004.
- 7 Scott Shaobing Chen, David L. Donoho, and Michael A. Saunders. Atomic Decomposition by Basis Pursuit. *SIAM Rev.*, 43(1):129–159, 2001.
- 8 Graham Cormode and S. Muthukrishnan. An improved data stream summary: the count-min sketch and its applications. *J. Algorithms*, 55(1):58–75, 2005. doi:10.1016/j.jalgor.2003.12.001.
- 9 Yevgeniy Dodis, Rafail Ostrovsky, Leonid Reyzin, and Adam Smith. Fuzzy Extractors: How to Generate Strong Keys from Biometrics and Other Noisy Data. *SIAM J. Comput.*, 38(1):97–139, March 2008. doi:10.1137/060651380.
- 10 Bella Dubrov and Yuval Ishai. On the randomness complexity of efficient sampling. In *Proceedings of the 38th Annual ACM Symposium on Theory of Computing, Seattle, WA, USA, May 21-23, 2006*, pages 711–720, 2006.
- 11 Oded Goldreich, Shafi Goldwasser, and Silvio Micali. How to construct random functions. *J. ACM*, 33(4):792–807, 1986.
- 12 Moritz Hardt and David P. Woodruff. How robust are linear sketches to adaptive inputs? In Dan Boneh, Tim Roughgarden, and Joan Feigenbaum, editors, *Symposium on Theory of Computing Conference, STOC'13, Palo Alto, CA, USA, June 1-4, 2013*, pages 121–130. ACM, 2013. doi:10.1145/2488608.2488624.
- 13 Piotr Indyk. Stable Distributions, Pseudorandom Generators, Embeddings and Data Stream Computation. In *41st Annual Symposium on Foundations of Computer Science, FOCS 2000, 12-14 November 2000, Redondo Beach, California, USA*, pages 189–197, 2000.
- 14 Piotr Indyk and Rajeev Motwani. Approximate Nearest Neighbors: Towards Removing the Curse of Dimensionality. In *Proceedings of the Thirtieth Annual ACM Symposium on the Theory of Computing, Dallas, Texas, USA, May 23-26, 1998*, pages 604–613, 1998.
- 15 T. S. Jayram, Ravi Kumar, and D. Sivakumar. The One-Way Communication Complexity of Hamming Distance. *Theory of Computing*, 4(1):129–135, 2008. doi:10.4086/toc.2008.v004a006.
- 16 Harini Kannan, Alexey Kurakin, and Ian J. Goodfellow. Adversarial Logit Pairing. *CoRR*, abs/1803.06373, 2018. arXiv:1803.06373.
- 17 J. Zico Kolter and Eric Wong. Provable defenses against adversarial examples via the convex outer adversarial polytope. *CoRR*, abs/1711.00851, 2017. arXiv:1711.00851.
- 18 Ilan Kremer, Noam Nisan, and Dana Ron. On Randomized One-round Communication Complexity. In *Proceedings of the Twenty-seventh Annual ACM Symposium on Theory of Computing, STOC '95*, pages 596–605, New York, NY, USA, 1995. ACM. doi:10.1145/225058.225277.
- 19 Eyal Kushilevitz, Rafail Ostrovsky, and Yuval Rabani. Efficient Search for Approximate Nearest Neighbor in High Dimensional Spaces. In *Proceedings of the Thirtieth Annual ACM Symposium on Theory of Computing, STOC '98*, pages 614–623, New York, NY, USA, 1998. ACM. doi:10.1145/276698.276877.
- 20 Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards Deep Learning Models Resistant to Adversarial Attacks. *CoRR*, abs/1706.06083, 2017. arXiv:1706.06083.
- 21 Ilya Mironov, Moni Naor, and Gil Segev. Sketching in adversarial environments. In *Proceedings of the 40th Annual ACM Symposium on Theory of Computing, Victoria, British Columbia, Canada, May 17-20, 2008*, pages 651–660, 2008.

- 22 Jayadev Misra and David Gries. Finding Repeated Elements. *Sci. Comput. Program.*, 2(2):143–152, 1982.
- 23 J. Ian Munro and Mike Paterson. Selection and Sorting with Limited Storage. *Theor. Comput. Sci.*, 12:315–323, 1980.
- 24 Moni Naor and Eylon Yogev. Bloom Filters in Adversarial Environments. In *Advances in Cryptology - CRYPTO 2015 - 35th Annual Cryptology Conference, Santa Barbara, CA, USA, August 16-20, 2015, Proceedings, Part II*, pages 565–584, 2015.
- 25 Moni Naor and Moti Yung. Universal One-Way Hash Functions and their Cryptographic Applications. In *Proceedings of the 21st Annual ACM Symposium on Theory of Computing, May 14-17, 1989, Seattle, Washington, USA*, pages 33–43, 1989.
- 26 Aditi Raghunathan, Jacob Steinhardt, and Percy Liang. Certified Defenses against Adversarial Examples. *CoRR*, abs/1801.09344, 2018. [arXiv:1801.09344](https://arxiv.org/abs/1801.09344).
- 27 Sivaramakrishnan Natarajan Ramamoorthy and Makrand Sinha. On the communication complexity of greater-than. In *53rd Annual Allerton Conference on Communication, Control, and Computing, Allerton 2015, Allerton Park & Retreat Center, Monticello, IL, USA, September 29 - October 2, 2015*, pages 442–444, 2015.
- 28 Aman Sinha, Hongseok Namkoong, and John C. Duchi. Certifiable Distributional Robustness with Principled Adversarial Training. *CoRR*, abs/1710.10571, 2017. [arXiv:1710.10571](https://arxiv.org/abs/1710.10571).
- 29 David Woodruff. Optimal Space Lower Bounds for All Frequency Moments. In *Proceedings of the Fifteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '04, pages 167–175, Philadelphia, PA, USA, 2004. Society for Industrial and Applied Mathematics. URL: <http://dl.acm.org/citation.cfm?id=982792.982817>.
- 30 David P. Woodruff. *Efficient and private distance approximation in the communication and streaming models*. PhD thesis, Massachusetts Institute of Technology, Cambridge, MA, USA, 2007. URL: <http://hdl.handle.net/1721.1/42243>.
- 31 Andrew Chi-Chih Yao. Some Complexity Questions Related to Distributive Computing (Preliminary Report). In *Proceedings of the 11th Annual ACM Symposium on Theory of Computing, April 30 - May 2, 1979, Atlanta, Georgia, USA*, pages 209–213, 1979.




# On Closest Pair in Euclidean Metric: Monochromatic is as Hard as Bichromatic

Karthik C. S.<sup>1</sup>

Weizmann Institute of Science, Rehovot, Israel

karthik.srikanta@weizmann.ac.il

 <https://orcid.org/0000-0001-9105-364X>

Pasin Manurangsi<sup>2</sup>

University of California, Berkeley, USA

pasin@berkeley.edu

---

## Abstract

Given a set of  $n$  points in  $\mathbb{R}^d$ , the (monochromatic) *Closest Pair* problem asks to find a pair of distinct points in the set that are closest in the  $\ell_p$ -metric. Closest Pair is a fundamental problem in Computational Geometry and understanding its fine-grained complexity in the Euclidean metric when  $d = \omega(\log n)$  was raised as an open question in recent works (Abboud-Rubinfeld-Williams [FOCS'17], Williams [SODA'18], David-Karthik-Laekhanukit [SoCG'18]).

In this paper, we show that for every  $p \in \mathbb{R}_{\geq 1} \cup \{0\}$ , under the Strong Exponential Time Hypothesis (SETH), for every  $\varepsilon > 0$ , the following holds:

- No algorithm running in time  $O(n^{2-\varepsilon})$  can solve the Closest Pair problem in  $d = (\log n)^{\Omega_\varepsilon(1)}$  dimensions in the  $\ell_p$ -metric.
- There exists  $\delta = \delta(\varepsilon) > 0$  and  $c = c(\varepsilon) \geq 1$  such that no algorithm running in time  $O(n^{1.5-\varepsilon})$  can approximate Closest Pair problem to a factor of  $(1 + \delta)$  in  $d \geq c \log n$  dimensions in the  $\ell_p$ -metric.

In particular, our first result is shown by establishing the computational equivalence of the *bichromatic* Closest Pair problem and the (monochromatic) Closest Pair problem (up to  $n^\varepsilon$  factor in the running time) for  $d = (\log n)^{\Omega_\varepsilon(1)}$  dimensions.

Additionally, under SETH, we rule out nearly-polynomial factor approximation algorithms running in subquadratic time for the (monochromatic) *Maximum Inner Product* problem where we are given a set of  $n$  points in  $n^{o(1)}$ -dimensional Euclidean space and are required to find a pair of distinct points in the set that maximize the inner product.

At the heart of all our proofs is the construction of a dense bipartite graph with low *contact dimension*, i.e., we construct a balanced bipartite graph on  $n$  vertices with  $n^{2-\varepsilon}$  edges whose vertices can be realized as points in a  $(\log n)^{\Omega_\varepsilon(1)}$ -dimensional Euclidean space such that every pair of vertices which have an edge in the graph are at distance exactly 1 and every other pair of vertices are at distance greater than 1. This graph construction is inspired by the construction of locally dense codes introduced by Dumer-Miccancio-Sudan [IEEE Trans. Inf. Theory'03].

**2012 ACM Subject Classification** Theory of computation → Computational geometry, Theory of computation → Problems, reductions and completeness

**Keywords and phrases** Closest Pair, Bichromatic Closest Pair, Contact Dimension, Fine-Grained Complexity

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2019.17

---

<sup>1</sup> Supported by Irit Dinur's ERC-CoG grant 772839 and BSF grant 2014371.

<sup>2</sup> Supported by NSF under Grants No. CCF 1655215 and CCF 1815434.



**Related Version** A full version of the paper is available at <https://arxiv.org/abs/1812.00901>.

**Acknowledgements** We are grateful to Madhu Sudan for extremely helpful and informative discussion about AG codes; in particular, Madhu pointed us to [54]. We thank Bundit Laekhanukit and Or Meir for general discussions, and the Simons Institute for their wonderful work-space. Finally, we would like to thank Lijie Chen for sharing [17], and Orr Paradise for useful comments on an earlier draft of this manuscript.

## 1 Introduction

The Closest Pair of Points problem or *Closest Pair* problem (CP) is a fundamental problem in computational geometry: given  $n$  points in a  $d$ -dimensional metric space, find a pair of distinct points with the smallest distance between them. The Closest Pair problem for points in the Euclidean plane [51, 11] stands at the origins of the systematic study of the computational complexity of geometric problems [45, 40, 36, 20]. Since then, this problem has found abundant applications in geographic information systems [28], clustering [58, 7], and numerous matching problems (such as stable marriage [56]).

The trivial algorithm for CP examines every pair of points in the point-set and runs in time  $O(n^2d)$ . Over the decades, there have been a series of developments on CP in low dimensional space for the Euclidean metric [10, 29, 35, 51, 11], leading to a deterministic  $O(2^{O(d)}n \log n)$ -time algorithm [11] and a randomized  $O(2^{O(d)}n)$ -time algorithm [46, 35]. For low (i.e., constant) dimensions, these algorithms are tight as a matching lower bound of  $\Omega(n \log n)$  was shown by Ben-Or [9] and Yao [57] in the *algebraic decision tree* model, thus settling the complexity of CP in low dimensions. On other hand, for very high dimensions (i.e.,  $d = \Omega(n)$ ) there are subcubic algorithms [27, 31] in the  $\ell_1, \ell_2$ , and  $\ell_\infty$ -metrics using fast matrix multiplication algorithms [25]. However, CP in medium dimensions, i.e.,  $d = \text{polylog}(n)$ , and in various  $\ell_p$ -metrics, have been a focus of study in machine learning and analysis of Big Data [37], and it is surprising that, even with the tools and techniques that have been developed over many decades, when  $d = \omega(\log n)$ , there is no known subquadratic-time (i.e.,  $O(2^{o(d)}n^{2-\varepsilon})$ -time) algorithm, for CP in any standard distance measure [30, 4, 31]. The absence of such algorithms was explicitly observed as early as the late nineties by Cohen and Lewis [19] but there was not any explanation until recently.

David, Karthik, and Laekhanukit [21] showed that for all  $p > 2$ , assuming the *Strong Exponential Time Hypothesis* (SETH), for every  $\varepsilon > 0$ , no algorithm running in  $n^{2-\varepsilon}$  time can solve CP in the  $\ell_p$ -metric, even when  $d = \omega(\log n)$ . Their conditional lower bound was based on the conditional lower bound (again assuming SETH) of Alman and Williams [6] for the *Bichromatic Closest Pair* problem<sup>3</sup> (BCP) where we are given two sets of  $n$  points in a  $d$ -dimensional metric space, and the goal is to find a pair of points, one from each set, with the smallest distance between them. Alman and Williams showed that for all  $p \in \mathbb{R}_{\geq 1} \cup \{0\}$ , assuming SETH, for every  $\varepsilon > 0$ , no algorithm running in  $n^{2-\varepsilon}$  time can solve BCP in the  $\omega(\log n)$ -dimensional  $\ell_p$ -metric space. Given that [6] show their lower bound on BCP for all  $\ell_p$ -metrics, the lower bound on CP of [21] feels unsatisfactory, since the  $\ell_2$ -metric is arguably the most interesting metric to study CP on. On the other hand, the answer to

<sup>3</sup> We remark that BCP is of independent interest as it's equivalent to finding the *Minimum Spanning Tree* in  $\ell_p$ -metric [3, 38]. Moreover, understanding the fine-grained complexity of BCP has lead to better understanding of the query time needed for *Approximate Nearest Neighbor* search problem (see Razenshteyn's thesis [47] for a survey about the problem) with polynomial preprocessing time [50].

the complexity of CP in the Euclidean metric might be on the positive side, i.e., there might exist an algorithm that performs well in the  $\ell_2$ -metric because there are more tools available, e.g., Johnson-Lindenstrauss' dimension reduction [33]. Thus we have the following question:

► **Open Question 1** (Abboud-Rubinfeld-Williams<sup>4</sup> [2], Williams [55], David -Karthik-Laekhanukit [21]). *Is there an algorithm running in time  $n^{2-\varepsilon}$  for some  $\varepsilon > 0$  which can solve CP in the Euclidean metric when the points are in  $\omega(\log n)$  dimensions?*

Even if the answer to the above question is negative, this does not rule out strong approximation algorithms for CP in the Euclidean metric, which might suffice for all applications. Indeed, we do know of subquadratic approximation algorithms for CP. For example, LSH based techniques can solve  $(1 + \delta)$ -CP (i.e.,  $(1 + \delta)$  factor approximate CP) in  $n^{2-\Theta(\delta)}$  time [32], but cannot do much better [42, 43]. In a recent breakthrough, Valiant [52] obtained an approximation algorithm for  $(1 + \delta)$ -CP with runtime of  $n^{2-\Theta(\sqrt{\delta})}$ . The state of the art is an  $n^{2-\tilde{\Theta}(\delta^{1/3})}$ -time algorithm by Alman, Chan, and Williams [5]. Can the dependence on  $\delta$  be improved indefinitely? For the case of  $(1 + \delta)$ -BCP, assuming SETH, Rubinfeld [50] answered the question in the negative. Does  $(1 + \delta)$ -CP also admit the same negative answer?

► **Open Question 2.** *Is there an algorithm running in time  $n^{2-\varepsilon}$  for some  $\varepsilon > 0$  which can solve  $(1 + \delta)$ -CP in the Euclidean metric when the points are in  $\omega(\log n)$  dimensions for every  $\delta > 0$ ?*

Another important geometric problem is the *Maximum Inner Product* problem (MIP): given  $n$  points in the  $d$ -dimensional Euclidean space, find a pair of distinct points with the largest inner product. This problem along with its bichromatic variant (*Bichromatic Maximum Inner Product* problem, denoted BMIP) is extensively studied in literature (see [2] and references therein). Abboud, Rubinfeld, and Williams [2] showed that assuming SETH, for every  $\varepsilon > 0$ , no  $2^{(\log n)^{1-o(1)}}$ -approximation algorithm running in  $n^{2-\varepsilon}$  time can solve BMIP when  $d = n^{o(1)}$ . It is a natural question to ask if their inapproximability result can be extended to MIP:

► **Open Question 3.** *Is there an algorithm running in time  $n^{2-\varepsilon}$  for some  $\varepsilon > 0$  which can solve  $\gamma$ -MIP in  $n^{o(1)}$  dimensions for even  $\gamma = 2^{(\log n)^{1-o(1)}}$ ?*

## 1.1 Our Results

In this paper we address all three previously mentioned open questions. First, we almost completely resolve Open Question 1. In particular, we show the following.

► **Theorem 4** (Subquadratic Hardness of CP). *Let  $p \in \mathbb{R}_{\geq 1} \cup \{0\}$ . Assuming SETH, for every  $\varepsilon > 0$ , no algorithm running in  $n^{2-\varepsilon}$  time can solve CP in the  $\ell_p$ -metric, even when  $d = (\log n)^{\Omega_\varepsilon(1)}$ .*

In particular we would like to emphasize that the dimension for which we show the lower bound on CP depends on  $\varepsilon$ . We would also like to remark that our lower bound holds even when the input point-set of CP is a subset of  $\{0, 1\}^d$ . Finally, we note that the centerpiece of the proof of the above theorem (and also the proofs of the other results that will be subsequently mentioned) is the construction of a dense bipartite graph with low *contact dimension*, i.e., we construct a balanced bipartite graph on  $n$  vertices with  $n^{2-\varepsilon}$  edges whose

<sup>4</sup> Please see the erratum in [1].

vertices can be realized as points in a  $(\log n)^{\Omega_\varepsilon(1)}$ -dimensional  $\ell_p$ -metric space such that every pair of vertices which have an edge in the graph are at distance exactly 1 and every other pair of vertices are at distance greater than 1. This graph construction is inspired by the construction of locally dense codes introduced by Dumer, Miccancio, and Sudan [23] and uses special density properties of Reed Solomon codes. A detailed proof overview is given in Section 2.1.

Next, we improve our result in Theorem 4 in some aspects by showing  $1 + o(1)$  factor inapproximability of CP even in  $O_\varepsilon(\log n)$  dimensions, but can only rule out algorithms running in  $n^{1.5-\varepsilon}$  time (as opposed to Theorem 4 which rules out exact algorithms for CP running in  $n^{2-\varepsilon}$  time). More precisely, we show the following.

► **Theorem 5** (Subquadratic Hardness of gap-CP). *Let  $p \in \mathbb{R}_{\geq 1} \cup \{0\}$ . Assuming SETH, for every  $\varepsilon > 0$ , there exists  $\delta(\varepsilon) > 0$  and  $c(\varepsilon) > 1$  such that no algorithm running in  $n^{1.5-\varepsilon}$  time that can solve  $(1 + \delta)$ -CP in the  $\ell_p$ -metric, even when  $d = c \log n$ .*

We remark that the  $n^{1.5-\varepsilon}$  lower bound on approximate CP is an artifact of our proof strategy and that a different approach or an improvement in the state-of-the-art bound on the number of minimum weight codewords in algebraic geometric codes (which are used in our proof), will lead to the complete resolution of Open Question 2.

It should also be noted that the approximate version of CP and the dimension are closely related. Namely, using standard dimensionality reduction techniques [33]<sup>5</sup> for  $(1 + \delta)$ -CP, one can always assume that  $d = O_\delta(\log n)$ . In other words, hardness of  $(1 + \delta)$ -CP immediately yields logarithmic dimensionality bound as a byproduct.

Finally, we completely answer Open Question 3 by showing the following inapproximability result for MIP, matching the hardness for BMIP from [2].

► **Theorem 6** (Subquadratic Hardness of gap-MIP). *Assuming SETH, for every  $\varepsilon > 0$ , no algorithm running in  $n^{2-\varepsilon}$  time can solve  $\gamma$ -MIP for any  $\gamma \leq 2^{(\log n)^{1-o(1)}}$ , even when  $d = n^{o(1)}$ .*

Recently, there have been a lot of results connecting BCP or  $(1 + o(1))$ -BCP to other problems (see [50, 15, 16, 17]). Now such connections can be extended to CP as well. For example, the following conditional lower bound follows from [50] for gap-CP in the edit distance metric.

► **Theorem 7** (Subquadratic Hardness of gap-CP in edit distance metric). *Assuming SETH, for every  $\varepsilon > 0$ , there exists  $\delta(\varepsilon) > 0$  and  $c(\varepsilon) > 1$  such that no algorithm running in  $n^{1.5-\varepsilon}$  time can solve  $(1 + \delta)$ -CP in the edit distance metric, even when  $d = c \log n \log \log n$ .*

## 2 Proof Overview

In this section, we provide an overview of our proofs and the formal proofs may be found in the full version of the paper. For ease of presentation, we will sometimes be informal here; all notions and proofs are formalized in subsequent sections. Our overview is organized as follows. First, in Subsection 2.1, we outline our proof of running time lower bounds for exact CP (Theorem 4). Then, in Subsection 2.2, we abstract part of our reduction using error-correcting codes, and relate them back to the works on locally dense codes [23, 18, 41] that inspire our constructions. Finally, in Subsection 2.3, we briefly discuss how to modify the base construction (i.e. code properties) to give conditional lower bounds for approximate CP and MIP (Theorems 5 and 6).

<sup>5</sup> In fact, since our results applies to  $\{0, 1\}$ -vectors, simply subsampling coordinates would also work.

## 2.1 Conditional Lower Bound on Exact Closest Pair

In this subsection, we provide a proof overview of a slightly weaker version of Theorem 4, i.e., we show that assuming SETH, for every  $p \in \mathbb{R}_{\geq 1} \cup \{0\}$ , no subquadratic time algorithm can solve CP in the  $\ell_p$ -metric when  $d = (\log n)^{\omega(1)}$ . We prove such a result by reducing BCP in dimension  $d$  to CP in dimension  $d + (\log n)^{\omega(1)}$ , and the subquadratic hardness for CP follows from the subquadratic hardness of BCP established by [6]. Note that the results in this paper remain interesting even if SETH is false, as our reduction shows that BCP and CP are computationally equivalent<sup>6</sup> (up to  $n^{\omega(1)}$  factor in the running time) when  $d = (\log n)^{\omega(1)}$ . The conditional lower bound on CP is merely a consequence of this computational equivalence. Finally, we note that a similar equivalence also holds between MIP and BMIP.

**Understanding an obstacle of [21].** Our proof builds on the ideas of [21] who showed that assuming SETH, for every  $p > 2$ , no subquadratic time algorithm can solve CP in the  $\ell_p$ -metric when  $d = \omega(\log n)$ . They did so by connecting the complexity of CP and BCP via the *contact dimension* of the balanced complete bipartite graph (biclique), denoted by  $K_{n,n}$ . We elaborate on this below.

To motivate the idea behind [21], let us first consider the trivial reduction from BCP to CP: given an instance  $A, B$  of BCP, we simply output  $A \cup B$  as an instance of CP. This reduction fails because there is no guarantee on the distances of a pair of points both in  $A$  (or both in  $B$ ). That is, there could be two points  $\mathbf{a}, \mathbf{a}' \in A$  such that  $\|\mathbf{a} - \mathbf{a}'\|_p$  is much smaller than the optimum of BCP on  $A, B$ . If we simply solve CP on  $A \cup B$ , we might find such  $\mathbf{a}, \mathbf{a}'$  as the optimal pair but this does not give the answer to the original BCP problem. In order to circumvent this issue, one needs a gadget that “stretch” pairs of points both in  $A$  or both in  $B$  further apart while keeping the pairs of points across  $A$  and  $B$  close (and preserving the optimum of BCP on  $A, B$ ). It turns out that this notion corresponds exactly to the contact dimension of the biclique, which we define below.

► **Definition 8** (Contact Dimension [44]). For any graph  $G = (V, E)$ , a mapping  $\tau : V \rightarrow \mathbb{R}^d$  is said to *realize*  $G$  (in the  $\ell_p$ -metric) if for some  $\beta > 0$ , the following holds for every distinct vertices  $u, v$ :

$$\|\tau(u) - \tau(v)\|_p = \beta \text{ if } \{u, v\} \in E, \text{ and,} \tag{1}$$

$$\|\tau(u) - \tau(v)\|_p > \beta \text{ otherwise.} \tag{2}$$

The *contact dimension* (in the  $\ell_p$ -metric) of  $G$ , denoted by  $\text{cd}_p(G)$ , is the minimum  $d \in \mathbb{N}$  such that there exists  $\tau : V \rightarrow \mathbb{R}^d$  realizing  $G$  in the  $\ell_p$ -metric.

In this paper, we will be mainly interested in the contact dimension of bipartite graphs. Specifically, [21] only consider the contact dimension of the biclique  $K_{n,n}$ . Notice that a realization of biclique ensures that vertices on the same side are far from each other while vertices on different sides are close to each other preserving the optimum of BCP; these are exactly the desired properties of a gadget outlined above. Using this, [21] give a reduction from BCP to CP which shows that the two are computationally equivalent whenever  $d = \Omega(\text{cd}_p(K_{n,n}))$ , as follows.

<sup>6</sup> We can reduce an instance of CP to an instance of BCP by randomly partitioning the input set of CP instance into two, and the optimal closest pair of points will be in different sets with probability  $1/2$  (and this reduction can be made deterministic).

## 17:6 On Closest Pair in Euclidean Metric

Let  $A, B \subseteq \mathbb{R}^d$  each of cardinality  $n$  be an instance of BCP and let  $\tau : A \dot{\cup} B \rightarrow \mathbb{R}^{\text{cd}_p(K_{n,n})}$  be a map realizing the biclique  $(A \dot{\cup} B, A \times B)$  in the  $\ell_p$ -metric; we may assume w.l.o.g. that  $\beta = 1$ . Let  $\delta$  be the distance between any point in  $A$  and any point in  $B$  (i.e.,  $\delta$  is an upper bound on the optimum of BCP). Let  $\rho > 0$  be such that  $\|\tau(\mathbf{a}) - \tau(\mathbf{b})\|_p > 1 + \rho$  for all  $\mathbf{a} \in A, \mathbf{b} \in B$  (and this is guaranteed to exist by (2)). Moreover, let  $k > \delta/\rho$  be any sufficiently large number. Consider the point-sets  $\tilde{A}, \tilde{B} \subseteq \mathbb{R}^{d+\text{cd}_p(K_{n,n})}$  of cardinality  $n$  each defined as

$$\tilde{A} = \{\mathbf{a} \circ (k \cdot \tau(\mathbf{a})) \mid \mathbf{a} \in A\}, \quad \tilde{B} = \{\mathbf{b} \circ (k \cdot \tau(\mathbf{b})) \mid \mathbf{b} \in B\},$$

where  $\circ$  denotes the concatenation between two vectors and  $k \cdot \mathbf{x}$  denotes the usual scalar-vector multiplication (i.e. scaling  $\mathbf{x}$  up by a factor of  $k$ ). For brevity, we write  $\tilde{\mathbf{a}}$  and  $\tilde{\mathbf{b}}$  to denote  $\mathbf{a} \circ (k \cdot \tau(\mathbf{a}))$  and  $\mathbf{b} \circ (k \cdot \tau(\mathbf{b}))$  respectively.

We now argue that, if we can find the closest pair of points in  $\tilde{A} \cup \tilde{B}$ , then we also immediately solve BCP for  $(A, B)$ . More precisely, we claim that  $(\mathbf{a}^*, \mathbf{b}^*) \in A \times B$  is a bichromatic closest pair of  $(A, B)$  if and only if  $(\tilde{\mathbf{a}}^*, \tilde{\mathbf{b}}^*)$  is a closest pair of  $\tilde{A} \cup \tilde{B}$ .

To see that this is the case, observe that, for cross pairs  $(\tilde{\mathbf{a}}, \tilde{\mathbf{b}}) \in \tilde{A} \times \tilde{B}$ , (1) implies that the distance  $\|\tilde{\mathbf{a}} - \tilde{\mathbf{b}}\|_p$  is exactly  $(k^p + \|\mathbf{a} - \mathbf{b}\|_p^p)^{1/p}$ ; hence, among these pairs,  $(\tilde{\mathbf{a}}^*, \tilde{\mathbf{b}}^*)$  is a closest pair iff  $(\mathbf{a}^*, \mathbf{b}^*)$  is a bichromatic closest pair in  $A, B$ . Notice also that, since the bichromatic closest pair in  $A, B$  is of distance at most  $\delta$ , the closest pair in  $\tilde{A} \cup \tilde{B}$  is of distance at most  $(k^p + \delta^p)^{1/p} \leq k + \delta$ .

On the other hand, for pairs both from  $\tilde{A}$  or both from  $\tilde{B}$ , the distance must be at least  $k(1 + \rho)$ , which is more than  $k + \delta$  from our choice of  $k$ . As a result, these pairs cannot be a closest pair in  $\tilde{A} \cup \tilde{B}$ , and this concludes the sketch of the proof.

There are a couple of details that we have glossed over here: one is that the gap  $\rho$  cannot be too small (e.g.,  $\rho$  cannot be as small as  $1/2^n$ ) and the other is that we should be able to construct  $\tau$  efficiently. Nevertheless, these are typically not an issue.

[21] show that  $\text{cd}_p(K_{n,n}) = \Theta(\log n)$  when  $p > 2$  and that the realization can be constructed efficiently and with sufficiently large  $\rho$ . This implies the subquadratic hardness of CP (by reduction from BCP) in the  $\ell_p$ -metric for all  $p > 2$  and  $d = \omega(\log n)$ . However, it was known that  $\text{cd}_2(K_{n,n}) = \Theta(n)$  [24]. Thus, they could *not* extend their conditional lower bound to CP in the Euclidean metric<sup>7</sup> even when  $d = o(n)$ . In fact, this is a serious obstacle as it rules out many natural approaches to reduce BCP to CP in a black-box manner. Elaborating, the lower bound on  $\text{cd}_2(K_{n,n})$  rules out local gadget reductions which would replace each point with a composition of that point and a gadget with a small increase in the number of dimensions, as such gadgets can be used to construct a realization of  $K_{n,n}$  in the Euclidean metric in a low dimensional space, contradicting the lower bound on  $\text{cd}_2(K_{n,n})$ .

**Overcoming the Obstacle: Beyond Biclique.** We overcome the above obstacle by considering dense bipartite graphs, instead of the biclique. More precisely, we show that there exists a balanced bipartite graph  $G^* = (A^* \dot{\cup} B^*, E^*)$  on  $2n$  vertices such that  $|E^*| \geq n^{2-o(1)}$  and  $\text{cd}_p(G^*)$  is small (i.e.  $\text{cd}_p(G^*) \leq (\log n)^{\omega(1)}$ ). We give a construction of such a graph below but before we do so, let us briefly argue why this suffices to show that BCP and CP are computationally equivalent (up to  $n^{o(1)}$  multiplicative overhead in the running time) for dimension  $d = \Omega(\text{cd}_p(G^*))$ .

<sup>7</sup> Note that plugging in the bound on  $\text{cd}_2(K_{n,n})$  in the result of [21] yields that assuming SETH, no subquadratic in  $n$  running time algorithm can solve CP when  $d = \Omega(n)$ . This is not a meaningful lower bound as just the input size of CP when  $d = \Omega(n)$  is  $\Omega(n^2)$ .



Let us consider the same reduction which produces  $\tilde{A}, \tilde{B}$  as before, but instead of using a realization of the biclique, we use a realization  $\tau$  of  $G^*$ . This reduction is of course incorrect: if  $(\mathbf{a}^*, \mathbf{b}^*)$  is not an edge in  $G^*$ , then  $\|\tau(\mathbf{a}^*) - \tau(\mathbf{b}^*)\|_p$  could be large and, thus the corresponding pair of points  $(\tilde{\mathbf{a}}^*, \tilde{\mathbf{b}}^*) \in \tilde{A} \times \tilde{B}$ , may not be the closest pair. Nevertheless, we are not totally hopeless: if  $(\mathbf{a}^*, \mathbf{b}^*)$  is an edge, then we are in good shape and the reduction is correct.

With the above observation in mind, consider picking a random permutation  $\pi$  of  $A \cup B$  such that  $\pi(A) = A$  and  $\pi(B) = B$  and then initiate the above reduction with the map  $(\tau \circ \pi)$  instead of  $\tau$ . Note that  $\tau \circ \pi$  is simply a realization of an appropriate permutation  $G'$  of  $G^*$  (i.e.,  $G'$  is isomorphic to  $G^*$ ). Due to this, the probability that we are “lucky” and  $(\mathbf{a}^*, \mathbf{b}^*)$  is an edge in  $G'$  is  $p := |E|/n^2$ ; when this is the case, solving CP on the resulting instance would give the correct answer for the original BCP instance. If we repeat this  $\log n/p = n^{o(1)}$  times, we would find the optimum of the original BCP instance with high probability.

To recap, even when  $G^*$  is not a biclique, we can still use it to give a reduction from BCP to CP, except that the reduction produces multiple (i.e.  $\tilde{O}(n^2/|E^*|)$ ) instances of CP. We remark here that the reduction can be derandomized: we can deterministically (and efficiently) pick the permutations so that the permuted graphs covers  $K_{n,n}$ . As a minor digression, we would like to draw a parallel here with a recent work of Abboud, Rubinfeld, and Williams [2]. The obstacle raised in [21] is about the impossibility of certain kinds of many-one gadget reductions. We overcame it by designing a reduction from BCP to CP which not only increased the number of dimensions but also the number of points (by creating multiple instances of CP). This technique is also utilized in [2] where they showed the impossibility of Deterministic Distributed PCPs (Theorem I.2 in [2]) but then overcame that obstacle by using an advice (which is then enumerated over resulting in multiple instances) to build Non-deterministic Distributed PCPs.

**Constructing a dense bipartite graph with low contact dimension.** We now proceed to construct the desired graph  $G^* = (A^* \cup B^*, E^*)$ . Note that any construction of a dense bipartite graph with contact dimension  $n^{o(1)}$  is non-trivial. This is because it is known that a random graph has contact dimension  $\Omega(n)$  in the Euclidean metric with high probability [49, 13], and therefore our graph construction must be significantly better than a random graph.

Our realization  $\tau^*$  of  $G^*$  will map into a subset of  $\{0, 1\}^{(\log n)^{\omega(1)}}$ . As a result, we can fix  $p = 0$ , since a realization of a graph with entries in  $\{0, 1\}$  in the Hamming-metric also realizes the same graph in every  $\ell_p$ -metric for any  $p \neq \infty$ .

Fix  $g = \omega(1)$ . We associate  $[n]$  with  $\mathbb{F}_q^h$  where  $q = \Theta((\log n)^g)$  is a prime and  $h = \Theta\left(\frac{\log n}{g \cdot \log \log n}\right)$ . Let  $\mathcal{P}$  be the set of all univariate polynomials (in  $x$ ) over  $\mathbb{F}_q$  of degree at most  $h - 1$ . We have that  $|\mathcal{P}| = q^h = n$  and associate  $\mathcal{P}$  with  $A^*$ . Let  $\mathcal{Q}$  be the set of all univariate monic polynomials (in  $x$ ) over  $\mathbb{F}_q$  of degree  $h$ , i.e.,

$$\mathcal{Q} = \{x^h + p(x) \mid p(x) \in \mathcal{P}\}.$$

We associate the polynomials in  $\mathcal{Q}$  with the vertices in  $B^*$  (note that  $|\mathcal{Q}| = n$ ). In fact, we view the vertices in  $A^*$  and  $B^*$  as being uniquely labeled by polynomials in  $\mathcal{P}$  and  $\mathcal{Q}$  respectively. For notational clarity, we write  $p_a$  (resp.  $p_b$ ) to denote the polynomial in  $\mathcal{P}$  (resp.  $\mathcal{Q}$ ) that is associated to  $a \in A^*$  (resp.  $b \in B^*$ ).

For every  $a \in A^*$  and  $b \in B^*$ , we include  $(a, b)$  as an edge in  $E^*$  if and only if the polynomial  $p_b - p_a$  (which is of degree  $h$ ) has  $h$  distinct roots. This completes the construction of  $G^*$ . We have to now show the following two claims about  $G^*$ : (i)  $|E^*| = n^{2-O(1/g)} = n^{2-o(1)}$  and (ii) there is  $\tau : A^* \dot{\cup} B^* \rightarrow \{0, 1\}^{(\log n)^{O(g)}} = \{0, 1\}^{(\log n)^{\omega(1)}}$  that realizes  $G^*$ .

To show (i), let  $\mathcal{R}$  be the set of all monic polynomials of degree  $h$  with  $h$  distinct roots. We have that  $|\mathcal{R}| = \binom{q}{h}$ . Fix a vertex  $a \in A^*$ . Its degree in  $G^*$  is exactly  $|\mathcal{R}| = \binom{q}{h}$ . This is because, for every polynomial  $r \in \mathcal{R}$ ,  $r + a$  belongs to  $\mathcal{Q}$ , and therefore  $(a, r + a) \in E^*$ . This implies the following bound on  $|E^*|$ :

$$|E^*| = q^h \cdot \binom{q}{h} \geq q^h \cdot \frac{q^h}{h^h} > \frac{n^2}{(\log n)^{\Theta((\log n)/(g \cdot \log \log n))}} = n^{2-O(1/g)}.$$

Next, to show (ii), we construct a realization  $\tau^* : A^* \dot{\cup} B^* \rightarrow \mathbb{F}_q^q$  of  $G^*$ . We note that, it is simple to translate the entries to  $\{0, 1\}$  instead of  $\mathbb{F}_q$ , by replacing  $i \in \mathbb{F}_q$  with the  $i$ -th standard basis  $\mathbf{e}_i \in \{0, 1\}^q$ . This would result in a realization  $\tau^* : A^* \dot{\cup} B^* \rightarrow \{0, 1\}^{q^2}$  of  $G^*$ ; notice that the dimension of  $\tau^*$  is  $q^2 = \Theta((\log n)^{2g})$  as claimed.

We define  $\tau^*$  as follows.

- For every  $a \in A^*$ ,  $\tau^*(a)$  is simply the vector of evaluation of  $p_a$  on every element in  $\mathbb{F}_q$ . More precisely, for every  $j \in [q]$ , the  $j$ -th coordinate of  $\tau^*(a)$  is  $p_a(j - 1)$ .
- Similarly, for every  $b \in B^*$  and  $j \in [q]$ , the  $j$ -th coordinate of  $\tau^*(b)$  is  $p_b(j - 1)$ .

We now show that  $\tau^*$  is indeed a realization of  $G^*$ ; specifically, we show that  $\tau^*$  satisfies (1) and (2) with  $\beta = q - h$ .

Consider any edge  $(a, b) \in E^*$ . Notice that  $\|\tau^*(a) - \tau^*(b)\|_0$  is the number of  $x \in \mathbb{F}_q$  such that  $p_b(x) - p_a(x) \neq 0$ . By definition of  $E^*$ ,  $p_b - p_a$  is a polynomial with  $h$  distinct roots over  $\mathbb{F}_q$ . Thus,  $\|\tau^*(a) - \tau^*(b)\|_0 = q - h = \beta$  as desired.

Next, consider a non-edge  $(a, b) \in (A^* \times B^*) \setminus E^*$ . Then, we know that  $p_b - p_a$  has at most  $h - 1$  distinct roots over  $\mathbb{F}_q$ . Therefore, the polynomial  $p_b - p_a$  is non-zero on at least  $q - h + 1$  coordinates. This implies that  $\|\tau^*(a) - \tau^*(b)\|_0 \geq q - h + 1 > \beta$ .

Finally, for any distinct  $a, a' \in A^*$ , we have  $\|\tau^*(a) - \tau^*(a')\|_0 \geq q - h + 1$  because  $p_a - p_{a'}$  is a non-zero polynomial of degree at most  $h - 1$  and thus can be zero over  $\mathbb{F}_q$  in at most  $h - 1$  locations. Similarly,  $\|\tau^*(b) - \tau^*(b')\|_0 \geq q - h + 1$  for any distinct  $b, b' \in B^*$ .

This completes the proof sketch for both the claims about  $G^*$  and yields Theorem 4 for  $d = (\log n)^{\omega(1)}$ . Finally we remark that in the actual proof of Theorem 4, we will set the parameters in the above construction more carefully and achieve the bound  $\text{cd}_p(G^*) = (\log n)^{O_\varepsilon(1)}$ .

## 2.2 Abstracting the Construction via Error-Correcting Codes

Before we move on to discuss the proofs of Theorems 6 and 5, let us give an abstraction of the construction in the previous subsection. This will allow us to easily generalize the construction for the aforementioned theorems, and also to explain where our motivation behind the construction comes from in the first place.

**Dense Bipartite Graph with Low Contact Dimension from Codes.** In order to construct a balanced bipartite graph  $G^*$  on  $2n$  vertices with  $n^{2-o(1)}$  edges such that  $\text{cd}_p(G^*) \leq d^*$ , it suffices to have a code  $C^*$  with the following properties:

- $C^* \subseteq \mathbb{F}_q^\ell$  of cardinality  $n$  is a linear code with block length  $\ell$  over alphabet  $\mathbb{F}_q$ , and minimum distance  $\Delta$ .
- There exists a *center*  $s^* \in \mathbb{F}_q^\ell$  and  $r^* < \Delta$  such that  $|C^*|^{1-o(1)}$  codewords are at Hamming distance exactly  $r^*$  from  $s^*$  and no codeword is at distance less than  $r^*$  from  $s^*$ .
- $q \cdot \ell = d^*$ .



We also require that  $C^*$  and  $s^*$  can be constructed in  $\text{poly}(n)$  time but we shall ignore this requirement for the ease of exposition.

We describe below how to construct  $G^*$  from  $C^*$ , but first note that the construction of  $G^*$  we saw in the previous subsection was just showing that Reed Solomon codes [48] of block length  $q = \Theta((\log n)^g)$  and message length  $h = \Theta\left(\frac{\log n}{g \cdot \log \log n}\right)$  over alphabet  $\mathbb{F}_q$  with minimum distance  $q - h + 1$  has the above properties. The center  $s^*$  in that construction was the evaluation of the polynomial  $x^h$  over  $\mathbb{F}_q$ , and  $r^*$  was  $q - h$ .

In general, to construct  $G^*$  from  $C^*$ , we first define a subset  $S^* \subseteq \mathbb{F}_q^\ell$  of cardinality  $n$  as follows:

$$S^* = \{s^* + \mathbf{c} \mid \mathbf{c} \in C^*\}.$$

We associate the vertices in  $A^*$  with the codewords of  $C^*$  and vertices in  $B^*$  with the strings in  $S^*$ . For any  $(\mathbf{a}, \mathbf{b}) \in A^* \times B^*$ , let  $(\mathbf{a}, \mathbf{b}) \in E^*$  if and only if  $\|\mathbf{b} - \mathbf{a}\|_0 = r^*$ . This completes the construction of  $G^*$ . We have to now show the following claims about  $G^*$ : (i)  $|E^*| = n^{2-o(1)}$  and (ii) there is  $\tau : A^* \dot{\cup} B^* \rightarrow \{0, 1\}^{q \cdot \ell}$  that realizes  $G^*$ .

Item (i) follows rather easily from the properties of  $C^*$  and  $s^*$ . Let  $T^*$  be the subset of  $C^*$  of all codewords which are at distance exactly equal to  $r^*$  from  $s^*$ . From the definition of  $s^*$ , we have  $|T^*| = |C^*|^{1-o(1)}$ . Fix  $\mathbf{a} \in A^*$ . Its degree in  $G^*$  is  $|T^*| = |C^*|^{1-o(1)}$ . This is because for every codeword  $\mathbf{t} \in T^*$  we have that  $\mathbf{t} - \mathbf{a}$  is a codeword in  $C^*$  (from the linearity of  $C^*$ ) and thus  $s^* - \mathbf{t} + \mathbf{a}$  is in  $S^*$ , and therefore  $(\mathbf{a}, s^* - \mathbf{t} + \mathbf{a}) \in E^*$ .

For item (ii), consider the identity mapping  $\tau^* : A^* \dot{\cup} B^* \rightarrow \mathbb{F}_q^\ell$  that maps each string to itself. It is simple to check that  $\tau^*$  realizes  $G^*$  in the Hamming metric (with  $\beta = r^*$ ).

Recall from the previous subsection that given  $\tau^* : A^* \dot{\cup} B^* \rightarrow \mathbb{F}_q^\ell$  that realizes  $G^*$  in the Hamming metric, it is easy to construct  $\tau : A^* \dot{\cup} B^* \rightarrow \{0, 1\}^{q \cdot \ell}$  that realizes  $G^*$  in the Hamming metric with a  $q$  multiplicative factor blow-up in the dimension. This completes the proof of both the claims about  $G^*$  and gives a general way to prove Theorem 4 given the construction of  $C^*$  and  $s^*$ .

**Finding Center from Another Code.** One thing that might not be clear so far is: where does the center  $s^*$  come from? Here we provide a systematic way to produce such an  $s^*$ , by looking at another code that contains  $C^*$ . More precisely, let  $C^* \subseteq \tilde{C}^* \subseteq \mathbb{F}_q^\ell$  be two linear codes with the same block length and alphabet. Suppose that the distance of  $C^*$  is  $\Delta$ , the distance of  $\tilde{C}^*$  is  $r^*$  and that  $r^* < \Delta$ . It is easy to see that, by taking  $s^*$  to be any element of  $\tilde{C}^* \setminus C^*$ , it holds that every codeword in  $C^*$  is at distance at least  $r^*$  from  $s^*$ , simply because both  $s^*$  and the codewords of  $C^*$  are codewords of  $\tilde{C}^*$ .

Hence, we are only left to argue that there are many codewords of  $C^*$  that is of distance exactly  $r^*$  from  $s^*$ . While this is not true in general, we can show by an averaging argument that this is true (for some  $s^* \in \tilde{C}^*$ ) if a large fraction (e.g.  $|C^*|^{-o(1)}$  fraction) of codewords of  $\tilde{C}^*$  has Hamming weight exactly  $r^*$ .

Indeed, viewing in this light, our previous choice of center for Reed-Solomon code (i.e. evaluation of  $x^h$ ) is not coincidental: we simply take  $\tilde{C}^*$  to be another Reed-Solomon code with message length  $h + 1$  (whereas the base code  $C^*$  is of message length  $h$ ).

**Comparison to Locally Dense Codes.** We end this subsection by remarking that the codes that we seek are very similar to locally dense codes [23, 18, 41], which is indeed our inspiration. A *locally dense code* is a linear code of block length  $\ell$  and large minimum distance

$\Delta$ , admitting a ball centered at  $s$  of radius<sup>8</sup>  $r < \Delta$  and containing a large (i.e.  $\exp(\text{poly}(\ell))$ ) number of codewords<sup>9</sup>. Such codes are non-trivial to construct and in particular all known constructions of locally dense codes are using codes that beat the Gilbert-Varshamov (GV) bound [26, 53]; in other words we need to do better than random codes to construct them. This is because (as noted in [23]), for a random code  $C \subseteq \mathbb{F}_q^\ell$  (or any code that does not beat the GV bound), a random point in  $\mathbb{F}_q^\ell$  acting as the center contains in expectation less than one codeword in a ball of radius  $\Delta$ . Of course, this is simply an intuition and not a formal proof that a locally dense code needs to beat the GV bound, since there may be more sophisticated ways to pick a center.

Although the codes we require are similar to locally dense codes, there are differences between the two. Below we list four such differences: the first two makes it *harder* for us to construct our codes whereas the latter two makes it *easier* for us.

- We seek a center  $s^*$  so that no codewords in  $C^*$  lies at distance less than  $r^*$ , as opposed to locally dense codes which allows codewords to be close to  $s^*$ . This is indeed where our idea of using another code  $\tilde{C}^* \supseteq C^*$  comes in, as picking  $s^*$  from  $\tilde{C}^* \setminus C^*$  ensures us that no codeword of  $C^*$  is too close to  $s^*$ .
- Another difference is that we need the number of codewords at distance  $r^*$  from  $s^*$  to be very large, i.e.,  $|C^*|^{1-o(1)}$ , whereas locally dense codes allow for much smaller number of codewords. Indeed, the deterministic constructions from [18, 41] only yield the bound of  $2^{\mathcal{O}(\sqrt{\log |C^*|})}$ . Hence, these do not directly work for us.
- Locally dense codes requires  $r$  to be at most  $(1 - \varepsilon)\Delta$  for some constant  $\varepsilon > 0$ , whereas we are fine with any  $r^* < \Delta$ . In fact, our Reed-Solomon code based construction above only yields  $r^* = \Delta - 1$  which would not suffice for locally dense codes. Nevertheless, as we will see later for inapproximability of CP, we will also need the ratio  $r^*/\Delta$  to be a constant bounded away from 1 as well and, since we need a code with these extraordinary properties, they are very hard to find. Indeed, in this case we only manage to prove a weaker lower bound on gap-CP.
- Finally, we remark that locally dense codes are required to be efficiently constructed in  $\text{poly}(\log |C^*|)$  time, which is part of why it is hard to find. Specifically, while [23] shows that an averaging argument works for a random center, derandomizing this is a big issue and a few subsequent works are dedicated solely to this issue [18, 41]. (We also note that it remains open whether a center can be deterministically found for a variant of locally dense codes used in hardness of parameterized version of the minimum distance problem. See [12] for more details.) On the other hand, brute force search (over all codewords in  $\tilde{C}^*$ ) suffices to find a center for us, as we are allowed construction time of  $\text{poly}(|C^*|)$ .

### 2.3 Inapproximability of Closest Pair and Maximum Inner Product

In this subsection, we sketch our inapproximability results for MIP and CP. Both these results use the same reduction that we had from BCP to CP, except that we now need stronger properties from the gadget, i.e., the previously used notions of contact dimension does not suffice anymore. Below we sketch the required strengthening of the gadget properties and explain how to achieve them.

<sup>8</sup> Clearly, for the ball to contain more than a single codeword, it must be  $r \geq \Delta/2$ . Here we are interested in balls with radius not much bigger than that, say  $r < \gamma \cdot \Delta$  for some constant  $1/2 < \gamma < 1$ .

<sup>9</sup> Strictly speaking, a locally dense code also requires an auxiliary matrix  $T$  used to index these codewords. However, in previous works, finding  $T$  is typically not hard given the center  $s$ . Hence, we ignore  $T$  in our discussion here for the ease of exposition.

### 2.3.1 Approximate Maximum Inner Product

Observe that the gadget we construct for CP in Subsection 2.2 can also be written in terms of inner product as follows: there exists a dense balanced bipartite graph  $G^* = (A^* \dot{\cup} B^*, E^*)$ , a mapping  $\tau : A^* \dot{\cup} B^* \rightarrow \{0, 1\}^{q \cdot \ell}$  such that the following holds.

- (i) For all edges  $(a, b) \in E^*$ ,  $\langle \tau(a), \tau(b) \rangle = \ell - r^*$ .
- (ii) For all edges  $(a, b) \in (A^* \times B^*) \setminus E^*$ ,  $\langle \tau(a), \tau(b) \rangle < \ell - r^*$ .
- (iii) For all distinct  $a, b$  both from  $A^*$  or both from  $B^*$ ,  $\langle \tau(a), \tau(b) \rangle \leq \ell - \Delta$ .

Notice that we wrote the conditions above in a slightly different way than in previous subsections; previously in the contact dimension notation, (ii) and (iii) would be simply written together as: for all non-edge  $(a, b)$ ,  $\langle \tau(a), \tau(b) \rangle < \ell - r^*$ . This change is intentional, since, to get gap in our reductions, we only need a gap between the bounds in (i) and (iii) (but not in (ii)). In particular, to get hardness of approximating MIP, we require  $\frac{\ell - r^*}{\ell - \Delta}$  to be at least  $(1 + \varepsilon)$  for some  $\varepsilon > 0$ .

From our Reed-Solomon construction above,  $\ell - \Delta$  and  $\ell - r^*$  are exactly the message length of  $C^*$  minus one and the message length of  $\tilde{C}^*$  minus one respectively. Previously, we selected these two to be  $h$  and  $h + 1$ . Now to obtain the desired gap, we simply take the larger code  $\tilde{C}^*$  to be a Reed-Solomon code with larger (i.e.  $(1 + \varepsilon)h$ ) message length<sup>10</sup>.

Finally, we note that even with the above gadget, the reduction only gives a small (i.e.  $1 + o(1)$ ) factor hardness of approximating MIP. To boost the gap to near polynomial, we simply tensor the vectors with themselves.

### 2.3.2 Approximate Closest Pair

Once again, recall that we have the following gadget from Subsection 2.2: there exists a dense balanced bipartite graph  $G^* = (A^* \dot{\cup} B^*, E^*)$ , a mapping  $\tau : A^* \dot{\cup} B^* \rightarrow \{0, 1\}^{q \cdot \ell}$  such that the following holds.

- (i) For all edges  $(a, b) \in E^*$ ,  $\|\tau(a) - \tau(b)\|_0 = r^*$ .
- (ii) For all edges  $(a, b) \in (A^* \times B^*) \setminus E^*$ ,  $\|\tau(a) - \tau(b)\|_0 > r^*$ .
- (iii) For all distinct  $a, b$  both from  $A^*$  or both from  $B^*$ ,  $\|\tau(a) - \tau(b)\|_0 \geq \Delta$ .

Once again, we need an  $(1 + \varepsilon)$  gap between the bounds in (iii) and (i), i.e.,  $\frac{\Delta}{r^*}$ . Unfortunately, we cannot construct such codes using any of the Reed-Solomon code families. We turn to another type of codes that beat the Gilbert-Varshamov bound: Algebraic-Geometric (AG) codes. Similar to the Reed-Solomon code based construction, we take  $C^*$  as an AG code and  $\tilde{C}^*$  to be a “higher degree” AG code; getting the desired gap simply means that the distance of  $C^*$  must be at least  $(1 + \varepsilon)$  times the distance of  $\tilde{C}^*$ .

Recall from Subsection 2.2 also that, to bound the density of  $G^*$ , we need a lower bound on the number of minimum weight codewords of  $\tilde{C}^*$ . Such bounds for AG codes are non-trivial and we turn to the bounds from [8, 54]. Unfortunately, this only gives  $G^*$  with density  $|C^*|^{-1/2 - o(1)}$ , instead of  $|C^*|^{-o(1)}$  as before. This is indeed the reason that our running time lower bound for approximate CP is only  $n^{1.5 - \varepsilon}$ .

We are not aware of any result on the (asymptotic) tightness of the bounds from [8, 54] that we use. However, improving upon such bounds would have other consequences, such as a better bound on the kissing numbers of lattices constructed in [54]. As a result, it seems likely that more understanding of AG codes (and perhaps even new constructions) are needed in order to improve these bounds.

<sup>10</sup>This approach can in fact give not just  $(1 + \varepsilon)$  but arbitrarily large constant gap between the two cases. In the actual reduction, we take this gap to be 3, which makes some computations simpler.

### 3 Discussion and Open Questions

It remains open to completely resolve Open Questions 1 and 2. It is still possible that our framework can be used to resolve these problems: we just need to construct gadgets with better parameters! In particular, to resolve Question 2, it suffices to obtain codes which have a much larger fraction of minimum weight codewords than the state-of-the-art algebraic geometric codes while having the desirable properties of algebraic geometric codes (formalized below). This motivates us to ask the following purely coding theoretic question:

► **Open Question 9.** *For every  $0 < \delta < 1$ , are there linear codes  $\mathcal{C}_1 \subseteq \mathcal{C}_2 \subseteq \mathbb{F}_q^N$  both of block length  $N$  over alphabet  $\mathbb{F}_q$  such that the following holds:*

- $\Delta(\mathcal{C}_1) \geq (1 + f(\delta)) \cdot \Delta(\mathcal{C}_2)$ , for some  $f : (0, 1) \rightarrow (0, 1)$ .
- $|A_{\Delta(\mathcal{C}_2)}(\mathcal{C}_2)|/|\mathcal{C}_2| \geq |\mathcal{C}_1|^{-\delta}$ .

Apart from the aforementioned questions, Rubinfeld [50] pointed out an interesting obstacle, aptly dubbed the “triangle inequality barrier”, to obtain fine-grained lower bounds against 3-approximation algorithms for BCP (see Open Question 3 in [50]). In the case of CP, this barrier turns out to be against 2-approximation algorithms as noted in [21]. We reiterate this below as an open problem to be resolved:

► **Open Question 10.** *Can we show that assuming SETH, for some constant  $\varepsilon > 0$ , no algorithm running in time  $n^{1+\varepsilon}$  can solve 2-CP in any metric when the points are in  $\omega(\log n)$  dimensions?*

Another interesting direction is to extend the hardness of MIP to the  $k$ -vector generalization of the problem, called  $k$ -MIP. In  $k$ -MIP, we are given a set of  $n$  points  $P \subseteq \mathbb{R}^d$  and we would like to select  $k$  distinct points  $\mathbf{a}_1, \dots, \mathbf{a}_k \in P$  that maximizes

$$\langle \mathbf{a}_1, \dots, \mathbf{a}_k \rangle := \sum_{j \in [d]} (\mathbf{a}_1)_j \cdots (\mathbf{a}_k)_j.$$

It is known that the  $k$ -chromatic variant of  $k$ -MIP is hard to approximate (see Appendix B of [34]) but this is not known to be true for  $k$ -MIP itself. Our approach seems quite compatible to tackling this problem as well; in particular, if we can construct a certain (natural) generalization of our gadget for MIP, then we would immediately arrive at the inapproximability of  $k$ -MIP even for  $\{0, 1\}$ -entries vectors. The issue in constructing this gadget is that we are now concerned about agreements of more than two vectors, which does not correspond to error-correcting codes anymore and some additional tools are needed to argue for this more general case.

It should be noted that the hardness of approximating  $k$ -MIP for  $\{0, 1\}$ -entry vectors is equivalent to the *one-sided  $k$ -biclique* problem [39], in which a bipartite graph is given and the goal is to select  $k$  vertices on the right that maximize the number of their common neighbors. The equivalence can be easily seen by viewing the coordinates as the left-hand-side vertices and the vectors as the right-hand-side vertices. The one-sided  $k$ -biclique is shown to be  $W[1]$ -hard to approximate by Lin [39] who also showed a lower bound of  $n^{\Omega(\sqrt{k})}$  for the problem assuming ETH. If the generalization of our gadget for  $k$ -MIP works as intended, then this lower bound can be improved to  $n^{\Omega(k)}$  under ETH and even  $n^{k-o(1)}$  under SETH.

The one-sided  $k$ -biclique is closely related to the (two-sided)  $k$ -biclique problem, where we are given a bipartite graph and we wish to decide whether it contains  $K_{k,k}$  as a subgraph. The  $k$ -biclique problem was considered a major open problem in parameterized complexity (see e.g., [22]) until it was shown by Lin to be  $W[1]$ -hard [39]. Nevertheless, the running time

lower bound known is still not tight: currently, the best lower bound known for this problem is  $n^{\Omega(\sqrt{k})}$  both for the exact version (under ETH) [39] and its approximate variant (under Gap-ETH) [14]. It remains an interesting open question to close the gap between the above lower bounds and the trivial upper bound of  $n^{O(k)}$ . Progresses on the one-sided  $k$ -biclique problem could lead to improved lower bounds for  $k$ -biclique problem too, although several additional steps have to be taken care of.

---

## References

- 1 Amir Abboud, Aviad Rubinfeld, and Ryan Williams. Distributed PCP Theorems for Hardness of Approximation in P. *CoRR*, abs/1706.06407, 2017. [arXiv:1706.06407](#).
- 2 Amir Abboud, Aviad Rubinfeld, and Ryan Williams. Distributed PCP Theorems for Hardness of Approximation in P. In *FOCS*, pages 25–36, 2017.
- 3 Pankaj K. Agarwal, Herbert Edelsbrunner, Otfried Schwarzkopf, and Emo Welzl. Euclidean Minimum Spanning Trees and Bichromatic Closest Pairs. *Discrete & Computational Geometry*, 6:407–422, 1991. Preliminary version in SoCG’90. [doi:10.1007/BF02574698](#).
- 4 Nir Ailon and Bernard Chazelle. The Fast Johnson–Lindenstrauss Transform and Approximate Nearest Neighbors. *SIAM J. Comput.*, 39(1):302–322, 2009. Preliminary version in STOC’06. [doi:10.1137/060673096](#).
- 5 Josh Alman, Timothy M. Chan, and R. Ryan Williams. Polynomial Representations of Threshold Functions and Algorithmic Applications. In *IEEE 57th Annual Symposium on Foundations of Computer Science, FOCS 2016, 9-11 October 2016, Hyatt Regency, New Brunswick, New Jersey, USA*, pages 467–476, 2016. [doi:10.1109/FOCS.2016.57](#).
- 6 Josh Alman and Ryan Williams. Probabilistic Polynomials and Hamming Nearest Neighbors. In *IEEE 56th Annual Symposium on Foundations of Computer Science, FOCS 2015, Berkeley, CA, USA, 17-20 October, 2015*, pages 136–150, 2015. [doi:10.1109/FOCS.2015.18](#).
- 7 Ethem Alpaydin. *Introduction to Machine Learning*. The MIT Press, 2nd edition, 2010.
- 8 Alexei E. Ashikhmin, Alexander Barg, and Serge G. Vladut. Linear Codes with Exponentially Many Light Vectors. *J. Comb. Theory, Ser. A*, 96(2):396–399, 2001. [doi:10.1006/jcta.2001.3206](#).
- 9 Michael Ben-Or. Lower Bounds for Algebraic Computation Trees (Preliminary Report). In *Proceedings of the 15th Annual ACM Symposium on Theory of Computing, 25-27 April, 1983, Boston, Massachusetts, USA*, pages 80–86, 1983. [doi:10.1145/800061.808735](#).
- 10 Jon Louis Bentley. Multidimensional Divide-and-Conquer. *Commun. ACM*, 23(4):214–229, 1980. [doi:10.1145/358841.358850](#).
- 11 Jon Louis Bentley and Michael Ian Shamos. Divide-and-Conquer in Multidimensional Space. In *Proceedings of the 8th Annual ACM Symposium on Theory of Computing, May 3-5, 1976, Hershey, Pennsylvania, USA*, pages 220–230, 1976. [doi:10.1145/800113.803652](#).
- 12 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](#).
- 13 Yonatan Bilu and Nathan Linial. Monotone maps, sphericity and bounded second eigenvalue. *J. Comb. Theory, Ser. B*, 95(2):283–299, 2005. [doi:10.1016/j.jctb.2005.04.005](#).
- 14 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](#).

- 15 Lijie Chen. On The Hardness of Approximate and Exact (Bichromatic) Maximum Inner Product. In *33rd Computational Complexity Conference, CCC 2018, June 22-24, 2018, San Diego, CA, USA*, pages 14:1–14:45, 2018. doi:10.4230/LIPIcs.CCC.2018.14.
- 16 Lijie Chen. Toward Super-Polynomial Size Lower Bounds for Depth-Two Threshold Circuits. *CoRR*, abs/1805.10698, 2018. arXiv:1805.10698.
- 17 Lijie Chen and Ryan Williams. An Equivalence Class for Orthogonal Vectors. *To appear in SODA*, 2019.
- 18 Qi Cheng and Daqing Wan. A Deterministic Reduction for the Gap Minimum Distance Problem. *IEEE Trans. Information Theory*, 58(11):6935–6941, 2012. doi:10.1109/TIT.2012.2209198.
- 19 Edith Cohen and David D. Lewis. Approximating Matrix Multiplication for Pattern Recognition Tasks. *J. Algorithms*, 30(2):211–252, 1999. doi:10.1006/jagm.1998.0989.
- 20 Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. *Introduction to Algorithms, Third Edition*. The MIT Press, 3rd edition, 2009.
- 21 Roei David, Karthik C. S., and Bundit Laekhanukit. On the Complexity of Closest Pair via Polar-Pair of Point-Sets. In *34th International Symposium on Computational Geometry, SoCG 2018, June 11-14, 2018, Budapest, Hungary*, pages 28:1–28:15, 2018. doi:10.4230/LIPIcs.SoCG.2018.28.
- 22 Rodney G. Downey and Michael R. Fellows. *Fundamentals of Parameterized Complexity*. Texts in Computer Science. Springer, 2013. doi:10.1007/978-1-4471-5559-1.
- 23 Ilya Dumer, Daniele Micciancio, and Madhu Sudan. Hardness of approximating the minimum distance of a linear code. *IEEE Trans. Information Theory*, 49(1):22–37, 2003. doi:10.1109/TIT.2002.806118.
- 24 Peter Frankl and Hiroshi Maehara. On the Contact Dimensions of Graphs. *Discrete & Computational Geometry*, 3:89–96, 1988. doi:10.1007/BF02187899.
- 25 François Le Gall. Powers of tensors and fast matrix multiplication. In *International Symposium on Symbolic and Algebraic Computation, ISSAC'14, Kobe, Japan, July 23-25, 2014*, pages 296–303, 2014. doi:10.1145/2608628.2608664.
- 26 E. N. Gilbert. A comparison of signalling alphabets. *Bell System Technical Journal*, 31:504–522, 1952.
- 27 Omer Gold and Micha Sharir. Dominance Products and Faster Algorithms for High-Dimensional Closest Pair under  $\$L_\infty\$$ . *CoRR*, abs/1605.08107, 2016. arXiv:1605.08107.
- 28 Tomislav Hengl. Finding the right pixel size. *Computers & Geosciences*, 32(9):1283–1298, 2006. doi:10.1016/j.cageo.2005.11.008.
- 29 Klaus H. Hinrichs, Jürg Nievergelt, and Peter Schorn. Plane-Sweep Solves the Closest Pair Problem Elegantly. *Inf. Process. Lett.*, 26(5):255–261, 1988. doi:10.1016/0020-0190(88)90150-0.
- 30 Piotr Indyk. Dimensionality reduction techniques for proximity problems. In *Proceedings of the Eleventh Annual ACM-SIAM Symposium on Discrete Algorithms, January 9-11, 2000, San Francisco, CA, USA.*, pages 371–378, 2000. URL: <http://dl.acm.org/citation.cfm?id=338219.338582>.
- 31 Piotr Indyk, Moshe Lewenstein, Ohad Lipsky, and Ely Porat. Closest Pair Problems in Very High Dimensions. In *Automata, Languages and Programming: 31st International Colloquium, ICALP 2004, Turku, Finland, July 12-16, 2004. Proceedings*, pages 782–792, 2004. doi:10.1007/978-3-540-27836-8\_66.
- 32 Piotr Indyk and Rajeev Motwani. Approximate Nearest Neighbors: Towards Removing the Curse of Dimensionality. In *Proceedings of the Thirtieth Annual ACM Symposium on the Theory of Computing, Dallas, Texas, USA, May 23-26, 1998*, pages 604–613, 1998. doi:10.1145/276698.276876.



- 33 William Johnson and Joram Lindenstrauss. Extensions of Lipschitz mappings into a Hilbert space. In *Conference in modern analysis and probability (New Haven, Conn., 1982)*, volume 26 of *Contemporary Mathematics*, pages 189–206. American Mathematical Society, 1984.
- 34 Karthik C. S., Bundit Laekhanukit, and Pasin Manurangsi. On the Parameterized Complexity of Approximating Dominating Set. In *STOC*, 2018. To appear.
- 35 Samir Khuller and Yossi Matias. A Simple Randomized Sieve Algorithm for the Closest-Pair Problem. *Inf. Comput.*, 118(1):34–37, 1995. doi:10.1006/inco.1995.1049.
- 36 Jon Kleinberg and Éva Tardos. *Algorithm Design*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 2005.
- 37 Jon M. Kleinberg. Two Algorithms for Nearest-Neighbor Search in High Dimensions. In *Proceedings of the Twenty-Ninth Annual ACM Symposium on the Theory of Computing, El Paso, Texas, USA, May 4-6, 1997*, pages 599–608, 1997. doi:10.1145/258533.258653.
- 38 Drago Krzrnaric, Christos Levcopoulos, and Bengt J. Nilsson. Minimum Spanning Trees in  $d$  Dimensions. *Nord. J. Comput.*, 6(4):446–461, 1999.
- 39 Bingkai Lin. The Parameterized Complexity of  $k$ -Biclique. In *SODA*, pages 605–615, 2015.
- 40 Udi Manber. *Introduction to Algorithms: A Creative Approach*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1989.
- 41 Daniele Micciancio. Locally Dense Codes. In *IEEE 29th Conference on Computational Complexity, CCC 2014, Vancouver, BC, Canada, June 11-13, 2014*, pages 90–97, 2014. doi:10.1109/CCC.2014.17.
- 42 Rajeev Motwani, Assaf Naor, and Rina Panigrahy. Lower Bounds on Locality Sensitive Hashing. *SIAM J. Discrete Math.*, 21(4):930–935, 2007. doi:10.1137/050646858.
- 43 Ryan O’Donnell, Yi Wu, and Yuan Zhou. Optimal Lower Bounds for Locality-Sensitive Hashing (Except When  $q$  is Tiny). *TOCT*, 6(1):5:1–5:13, 2014. doi:10.1145/2578221.
- 44 Janos Pach. Decomposition of multiple packing and covering. *Diskrete Geometrie*, 2 Kolloq. Math. Inst. Univ. Salzburg:169–178, 1980.
- 45 Franco P. Preparata and Michael I. Shamos. *Computational Geometry: An Introduction*. Springer-Verlag New York, Inc., New York, NY, USA, 1985.
- 46 Michael O. Rabin. Probabilistic Algorithms. In *Proceedings of a Symposium on New Directions and Recent Results in Algorithms and Complexity, Computer Science Department, Carnegie-Mellon University, April 7-9, 1976*, pages 21–39, 1976.
- 47 Ilya Razenshteyn. High-Dimensional Similarity Search and Sketching: Algorithms and Hardness. *PhD Thesis, MIT*, 2017.
- 48 Irving S. Reed and Gustave Solomon. Polynomial Codes over Certain Finite Fields. *Journal of the Society for Industrial and Applied Mathematics (SIAM)*, 8(2):300–304, 1960.
- 49 Jan Reiterman, Vojtech Rödl, and Edita Sinajová. Embeddings of Graphs in Euclidean Spaces. *Discrete & Computational Geometry*, 4:349–364, 1989. doi:10.1007/BF02187736.
- 50 Aviad Rubinfeld. Hardness of approximate nearest neighbor search. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2018, Los Angeles, CA, USA, June 25-29, 2018*, pages 1260–1268, 2018. doi:10.1145/3188745.3188916.
- 51 Michael Ian Shamos and Dan Hoey. Closest-Point Problems. In *16th Annual Symposium on Foundations of Computer Science, Berkeley, California, USA, October 13-15, 1975*, pages 151–162, 1975. doi:10.1109/SFCS.1975.8.
- 52 Gregory Valiant. Finding Correlations in Subquadratic Time, with Applications to Learning Parities and the Closest Pair Problem. *J. ACM*, 62(2):13:1–13:45, 2015. doi:10.1145/2728167.
- 53 R. R. Varshamov. Estimate of the number of signals in error correcting codes. *Dokl. Akad. Nauk SSSR*, 117:739–741, 1957.

- 54 Serge Vlăduț. Lattices with exponentially large kissing numbers. *arXiv preprint*, 2018. [arXiv:1802.00886](https://arxiv.org/abs/1802.00886).
- 55 Ryan Williams. On the Difference Between Closest, Furthest, and Orthogonal Pairs: Nearly-Linear vs Barely-Subquadratic Complexity. In *Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2018, New Orleans, LA, USA, January 7-10, 2018*, pages 1207–1215, 2018. doi:10.1137/1.9781611975031.78.
- 56 Raymond Chi-Wing Wong, Yufei Tao, Ada Wai-Chee Fu, and Xiaokui Xiao. On Efficient Spatial Matching. In *Proceedings of the 33rd International Conference on Very Large Data Bases, University of Vienna, Austria, September 23-27, 2007*, pages 579–590, 2007. URL: <http://www.vldb.org/conf/2007/papers/research/p579-wong.pdf>.
- 57 Andrew Chi-Chih Yao. Lower Bounds for Algebraic Computation Trees with Integer Inputs. *SIAM J. Comput.*, 20(4):655–668, 1991. Preliminary version in FOCS’89. doi:10.1137/0220041.
- 58 Charles T. Zahn. Graph-Theoretical Methods for Detecting and Describing Gestalt Clusters. *IEEE Trans. Computers*, 20(1):68–86, 1971. doi:10.1109/T-C.1971.223083.



# Expander-Based Cryptography Meets Natural Proofs

**Igor C. Oliveira**

Department of Computer Science, University of Oxford, UK  
igor.carboni.oliveira@cs.ox.ac.uk

**Rahul Santhanam**

Department of Computer Science, University of Oxford, UK  
rahul.santhanam@cs.ox.ac.uk

**Roei Tell**

Department of Computer Science and Applied Mathematics, Weizmann Institute of Science, Rehovot, Israel  
roei.tell@weizmann.ac.il

---

## Abstract

We introduce new forms of attack on *expander-based cryptography*, and in particular on Goldreich’s pseudorandom generator and one-way function. Our attacks exploit *low circuit complexity* of the underlying expander’s neighbor function and/or of the local predicate. Our two key conceptual contributions are:

1. We put forward the possibility that the *choice of expander* matters in expander-based cryptography. In particular, using expanders whose neighbour function has low circuit complexity might compromise the security of Goldreich’s PRG and OWF in certain settings.
2. We show that the security of Goldreich’s PRG and OWF is closely related to two other long-standing problems: Specifically, to the existence of *unbalanced lossless expanders* with low-complexity neighbor function, and to *limitations on circuit lower bounds* (i.e., natural proofs). In particular, our results further motivate the investigation of affine/local unbalanced lossless expanders and of average-case lower bounds against DNF-XOR circuits.

We prove two types of technical results that support the above conceptual messages. First, we *unconditionally break Goldreich’s PRG* when instantiated with a specific expander (whose existence we prove), for a class of predicates that match the parameters of the currently-best “hard” candidates, in the regime of quasi-polynomial stretch. Secondly, conditioned on the existence of expanders whose neighbor functions have extremely low circuit complexity, we present attacks on Goldreich’s generator in the *regime of polynomial stretch*. As one corollary, conditioned on the existence of the foregoing expanders, we show that either the parameters of natural properties for several constant-depth circuit classes *cannot be improved, even mildly*; or Goldreich’s generator is insecure in the regime of a large polynomial stretch, *regardless of the predicate used*.

**2012 ACM Subject Classification** Theory of computation → Circuit complexity, Theory of computation → Cryptographic primitives, Theory of computation → Pseudorandomness and derandomization, Theory of computation → Expander graphs and randomness extractors

**Keywords and phrases** Pseudorandom Generators, One-Way Functions, Expanders, Circuit Complexity

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2019.18

**Related Version** For a full online version see [26], <https://eccc.weizmann.ac.il/report/2018/159/>.



© Igor C. Oliveira, Rahul Santhanam, and Roei Tell;  
licensed under Creative Commons License CC-BY

10th Innovations in Theoretical Computer Science (ITCS 2019).

Editor: Avrim Blum; Article No. 18; pp. 18:1–18:14



Leibniz International Proceedings in Informatics

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

**Acknowledgements** The authors thank Mahdi Cheraghchi, Emanuele Viola, and Avi Wigderson for useful e-mail exchanges about the complexity of constructing unbalanced lossless bipartite expander graphs. The authors are grateful to Benny Applebaum, Oded Goldreich, and Yuval Ishai for very helpful discussions about the implications of our work to expander-based cryptography. Finally, the authors thank anonymous reviewers for useful comments that improved the presentation of the paper. The first and second authors are supported by the second author’s ERC-CoG grant no. 615075. The third author is partially supported by Irit Dinur’s ERC-CoG grant no. 772839.

## 1 Introduction

Theoretical results provide strong evidence that if secure cryptography is possible, then many fundamental primitives such as one-way functions (OWF) and pseudorandom generators (PRG) can be implemented with a dramatic level of efficiency and parallelism. Specifically, security against efficient adversaries can be achieved by functions where each output bit only depends on a constant number of input bits (see, e.g., [6], and also [2] for a survey of recent results).

A concrete type of such construction is a conjectured form of OWF that is based on any *expander graph* and on a *local predicate*. Specifically, about two decades ago, Goldreich [16, 17] suggested the following candidate  $\text{owf} : \{0, 1\}^n \rightarrow \{0, 1\}^n$ . Fix any bipartite graph  $[n] \times [n]$  of right-degree  $\ell \leq O(\log(n))$  in which every set  $S \subseteq [n]$  of size up to  $k$  on the right-hand side has at least (say)  $1.01 \cdot |S|$  neighbors, and also fix a predicate  $P : \{0, 1\}^\ell \rightarrow \{0, 1\}$ . Then, given input  $x \in \{0, 1\}^n$ , each output bit  $\text{owf}(x)_i$  is computed by applying  $P$  to the bits of  $x$  at the  $\ell$  neighbors of  $i \in [n]$ . The expected running-time of a naive algorithm for inverting  $\text{owf}$  is at least  $\exp(k)$  (see, e.g., [17, Sec. 3.2] and [2, Sec. 3.1]), and Goldreich conjectured that for an appropriate predicate  $P$ , no algorithm can perform significantly better.

In an extensive subsequent line of research (see, e.g., [1, 23, 4, 9, 5, 10, 14, 25, 15, 7, 8], and also see [3] for a related survey), Goldreich’s construction was conjectured to yield not only a one-way function, but also a pseudorandom generator  $\text{prg} : \{0, 1\}^n \rightarrow \{0, 1\}^m$ . In fact, in some settings the two conjectures are essentially equivalent (see [8, Sec. 3]).

The question of whether Goldreich’s constructions are secure is a long-standing open problem. Much research has focused on necessary requirements from the *predicate* and from the *parameters* in order for the construction to be secure. Let us, for simplicity of presentation, focus on the PRG. In this case, the locality  $\ell$  cannot be too small: If we want a PRG with super-linear stretch, then we must use  $\ell \geq 5$  [23];<sup>1</sup> and if we want stretch  $m = n^k$  then  $\ell$  must be at least (roughly)  $3k$  (see [25, Thm. II.11]). Also, as shown in [7], the predicate must have *high resilience* (i.e., all of the predicate’s Fourier coefficients corresponding to sets of size at most  $\Omega(\ell)$  are zero; see [26, Def. 3.4]) and high *rational degree* (this is a generalization of the requirement that the degree of the predicate as a polynomial  $\mathbb{F}_2^\ell \rightarrow \mathbb{F}_2$  is  $\Omega(\ell)$ ; see [26, Def. 3.7]).

The foregoing properties capture most existing attacks in the PRG setting. Indeed, as mentioned above, all these attacks exploit vulnerabilities of the *predicate* and of the *parameters*, but not of the underlying expander. In fact, prior to our work, the PRG was conjectured to be secure for *any underlying expander* with sufficiently good expansion properties. For reference, let us state such a strong form of conjectured security of the OWF, from a recent work by Applebaum and Raykov [8]. We say that a bipartite graph

<sup>1</sup> This impossibility result holds for *any* construction of a pseudorandom generator in  $\mathcal{NC}^0$ .

$G = ([n], [m], E)$  with right-degree  $\ell$  is a  $(k, 0.99)$ -expander if for every set  $S \subseteq [m]$  on the right-hand side of size at most  $k$ , the number of neighbors of  $S$  in  $G$  is at least  $0.99 \cdot \ell \cdot |S|$ .<sup>2</sup> Then, the conjecture is the following:

► **Assumption 1** (the strong EOWF assumption). *For a family  $\mathcal{P} = \{P_\ell : \{0, 1\}^\ell \rightarrow \{0, 1\}\}_{\ell \in \mathbb{N}}$  of predicates, the strong EOWF( $\mathcal{P}$ ) assumption is the following. For any  $(n^{.99}, .99)$ -expander  $G = ([n], [m], E)$  of right-degree  $\ell \leq n^{o(1)}$  such that  $n \leq m \leq n^{\alpha \cdot \ell}$ , where  $\alpha > 0$  is a sufficiently small universal constant, Goldreich’s function instantiated with  $G$  and  $P_\ell$  cannot be inverted by circuits of size  $t \leq \exp(\alpha \cdot n^{.99})$  with success probability  $1/t$ .*

Applebaum and Raykov [8] suggested a suitable candidate predicate, which is the predicate  $\text{XOR-MAJ}(x) = (\oplus_{i=1, \dots, \lfloor \ell/2 \rfloor} x_i) \oplus (\text{MAJ}(x_{\lfloor \ell/2 \rfloor + 1}, \dots, x_\ell))$ ; this predicate indeed has both high resiliency and high rational degree.

## 1.1 A high-level digest of our contributions

Our main contribution is a *new form of attack* on Goldreich’s pseudorandom generator, which exploits *computational complexity* properties (and, in particular, circuit complexity properties) of the expander and/or of the predicate on which the generator is based. In particular, our distinguishers are algorithms associated with *natural properties*, in the sense of Razborov and Rudich [28]. (Recall that a natural property against a circuit class  $\mathcal{C}$  is an efficient algorithm that distinguishes a random string, interpreted as a truth table, from truth tables of  $\mathcal{C}$ -circuits.)<sup>3</sup>

We use our new form of attack to break the generator when it is instantiated with predicates that are sufficiently “strong” to withstand known attacks, but with expanders whose neighbor function has “low” circuit complexity. In high-level, the main conceptual implications of these results are the following:

1. The conjecture that the PRG and OWF are secure with *any expander*, given an appropriate predicate, *might be too naive*. In particular, the security of the constructions might crucially hinge on a choice of expander whose neighbor function has sufficiently high circuit complexity. Alternatively, if the latter is not true (i.e., if the PRG and OWF can be secure given any expander), then the predicate must have sufficiently high circuit complexity for the constructions to be secure in some settings (i.e., when the stretch is quasi-polynomial).

Note that a random graph will (with high probability) not only be an expander, but also have a neighbor function with high circuit complexity. Therefore, our results do not put into question the security of the PRG and OWF when instantiated with a random graph.

2. There are significant interdependencies between *the security of Goldreich’s PRG and OWF*, the existence of *unbalanced lossless expanders* with low-complexity neighbor function, and *limitations on circuit lower bounds* (i.e., natural proofs). Moreover (as further explained below), the questions motivated by our results are closely related both to existing results and to long-standing open problems in each area.

<sup>2</sup> We stress that lossless expansion (i.e., expansion to  $\alpha \cdot \ell \cdot |S|$  vertices for  $\alpha > 1/2$ ) is crucial in the PRG setting. To see this, note that one can duplicate a right-vertex in a  $(k, 0.99)$ -expander: This will produce a graph that, on the one hand, has good (but not lossless!) expansion properties, and on the other hand yields a corresponding PRG that is clearly insecure, regardless of the predicate.

<sup>3</sup> Natural properties are typically used to break *pseudorandom functions*, but the idea of using natural properties to break pseudorandom generators goes back to [28, Thm. 4.2]. Nevertheless, implementing this idea in our setting presents specific new challenges; for further discussion see Section 2.4.

Being more specific, we unconditionally break Goldreich’s generator in the setting of quasi-polynomial stretch when it is instantiated with predicates with *high resilience and rational degree*, but with an expander whose neighbor function can be computed by  $\mathcal{AC}^0[\oplus]$  circuits of (small) subexponential size. In fact, our predicates are variations on the specific XOR-MAJ predicate mentioned above. Using a known reduction of PRGs to OWFs (by [8]), it follows that Assumption 1 *does not* hold for some predicates with high resilience and rational degree. To prove this result we actually prove the *existence* of expanders with neighbor function as above; the latter proof, which uses certain *unconditional* PRGs that can be computed in a *strongly explicit* fashion, might be of independent interest. (See Section 1.2.)

In the regime of polynomial stretch, we put forward two assumptions about plausible extensions of known expander constructions in which the neighbor functions have even lower circuit complexity (compared to the expander mentioned above). Conditioned on *any* of the two assumptions, we show that exactly one of two options holds: Either the parameters of natural properties for certain restricted constant-depth circuit classes *cannot be improved, even mildly*; or Goldreich’s generator is insecure in the regime of a large polynomial stretch, *regardless of the predicate used*. (See Section 1.3.)

Some important cryptographic applications crucially rely on the security of expander-based PRGs with polynomial, or even linear, stretch (see, e.g., [3, Sec. 4, “The Stretch”] and the references therein). We stress that our results for the setting of polynomial stretch are conditional on the existence of suitable expanders, and only break the PRG and OWF if there are natural properties for constant-depth circuit classes beyond what is currently known. Thus, further investigation is needed to determine whether our results have implications on the security of the aforementioned applications.

## 1.2 Unconditional results for quasi-polynomial stretch

Our main result for the setting of quasi-polynomial stretch is an attack that *unconditionally breaks* Goldreich’s PRG when it is instantiated with a *specific expander that has optimal expansion properties*, and with a class of predicates that have both high resilience and high rational degree. Specifically:

► **Theorem 2** (unconditional attack on Goldreich’s PRG with quasi-polynomial stretch; informal). *For every  $d \in \mathbb{N}$  and sufficiently large  $k, c \in \mathbb{N}$  there exists a deterministic polynomial-time algorithm  $A$  that satisfies the following. Let  $n \in \mathbb{N}$  be sufficiently large, let  $m = n^{\log^k(n)}$ , and let  $\ell = c \cdot \log^k(n)$ . Then, there exists an  $(n^{0.99}, 0.99)$ -expander  $G = ([n], [m], E)$  of right-degree  $\ell$  such that for any predicate  $P : \{0, 1\}^\ell \rightarrow \{0, 1\}$  that can be computed by an  $\mathcal{AC}^0[\oplus]$  circuit of depth  $d$  and sufficiently small sub-exponential size, when Goldreich’s generator is instantiated with the expander  $G$  and the predicate  $P$ , the algorithm  $A$  distinguishes the  $m$ -bit output of the generator from a uniform  $m$ -bit string (with gap  $> 1/2$ ).*

In fact, we actually prove a more general theorem, which exhibits a trade-off between the locality  $\ell$  and the size of the  $\mathcal{AC}^0[\oplus]$  circuit for the predicate  $P$  (for a precise statement see [26, Thm. 4.6]). That is, we are able to break the generator even with much larger locality (e.g.,  $\ell = n^{0.01}$ ), at the expense of using a more restricted predicate family, namely that of  $\mathcal{AC}^0[\oplus]$  circuits of smaller size (e.g., polynomial size). We stress that even the latter predicate family is rich enough to contain predicates that have both high resilience and high rational degree (see below).

Recall that the property of the expander  $[n] \times [m]$  that we exploit in our attack is that its *neighbor functions* (i.e., the functions  $\Gamma_i : [m] \rightarrow [n]$  for  $i \in [\ell]$ ) have *low circuit complexity*. The expander in Theorem 2 in particular has neighbor functions that can be computed by  $\mathcal{AC}^0[\oplus]$  circuits of small sub-exponential size, and we prove its existence in [26, Sec. 4.1] (see Section 2.2 for a high-level description).

Combining Theorem 2 with Applebaum and Raykov’s reduction of expander-based PRGs to expander-based OWFs [8, Thm. 3.1] (i.e., they prove that if an arbitrary instance of Goldreich’s OWF is secure, then a closely-related instance of Goldreich’s PRG is also secure), our attack also breaks Goldreich’s OWF. Specifically, we say that a predicate  $P : \{0, 1\}^\ell \rightarrow \{0, 1\}$  is *sensitive* if it is “fully sensitive” to one of its coordinates (i.e., if for all  $x \in \{0, 1\}^\ell$  it holds that  $P(x) = x_i \oplus P'(x)$ , for some  $i \in [\ell]$  and  $P'$  that does not depend on  $x_i$ ). Then:

► **Corollary 3** (unconditional attack on Goldreich’s OWF with quasi-polynomial stretch; informal). *There exists a probabilistic polynomial-time algorithm  $A'$  that satisfies the following. Let  $n \in \mathbb{N}$  be sufficiently large, let  $m' = n^{k' = \text{poly} \log(n)}$ , and let  $\ell = O(k')$ . Then, there exists an  $(n^{0.99}, 0.99)$ -expander  $G = ([n], [m'], E)$  of right-degree  $\ell$  such that for any sensitive predicate  $P : \{0, 1\}^\ell \rightarrow \{0, 1\}$  that can be computed by an  $\mathcal{AC}^0[\oplus]$  circuit of sufficiently small sub-exponential size, when Goldreich’s one-way function is instantiated with the expander  $G$  and the predicate  $P$ , the algorithm  $A$  inverts the function with success probability  $\Omega(1/m'n)$ .*

As immediate corollaries of Theorem 2 and of Corollary 3, we deduce that Assumption 1 does not hold for any sensitive predicate family that can be computed by  $\mathcal{AC}^0[\oplus]$  circuits of sufficiently small sub-exponential size; and similarly, that the “PRG analogue” of Assumption 1, denoted  $EPRG(\mathcal{P})$  in [8], does not hold for any predicate family that can be computed by  $\mathcal{AC}^0[\oplus]$  circuits of sufficiently small sub-exponential size.

Recall that Applebaum and Raykov suggested the candidate predicate XOR-MAJ; we prove that when replacing majority by *approximate majority* (see [26, Def. 4.9]), the resulting predicate XOR-APPROX-MAJ still has both high resilience and high rational degree, and can also be computed by a polynomial-sized  $\mathcal{AC}^0[\oplus]$  circuit (see [26, Sec. 4.3.2]). Thus, the predicate families in Theorem 2 and Corollary 3 contain predicates with high resilience and high rational degree, and even predicates that are variations on the “hard” candidate XOR-MAJ.<sup>4</sup>

Moreover, the predicate XOR-APPROX-MAJ does not even use the “full power” of the predicate family for which Theorem 2 allows us to break Goldreich’s generator – the predicate XOR-APPROX-MAJ is computable by a circuit of polynomial size, whereas we can break the generator when the predicate can be computed by a circuit of sub-exponential size. We use this to our advantage by relying on the more general version of Theorem 2 (i.e., [26, Thm. 4.6]), which exhibits a trade-off between locality and the predicate class. Specifically, we obtain the following theorem, which breaks the generator even when the locality  $\ell$  is large (e.g.,  $\ell = n^{\Omega(1)}$ ) and the predicate has high resilience and rational degree:

► **Theorem 4** (breaking Goldreich’s generator with XOR-APPROX-MAJ and high locality). *There exists  $s > 1$  such that the following holds. Let  $n \in \mathbb{N}$ , let  $m = n^{k = (\log(n))^s}$ , and let  $c \cdot k \leq \ell \leq n^{1/c}$ , where  $c$  is a sufficiently large constant. Then, there exists an  $(n^{0.99}, 0.99)$ -expander  $G = ([n], [m], E)$  of right-degree  $\ell$  and a predicate  $P : \{0, 1\}^\ell \rightarrow \{0, 1\}$  with resilience  $\Omega(\ell)$  and rational degree  $\Omega(\ell)$  (i.e., the predicate XOR-APPROX-MAJ) such that the following holds: When Goldreich’s generator is instantiated with the expander  $G$  and the predicate  $P$ , the output of the generator can be distinguished from a uniform string (with gap  $> 1/2$ ) by a deterministic  $\text{poly}(m)$ -time algorithm.*

<sup>4</sup> Indeed, the main difference between XOR-MAJ and XOR-APPROX-MAJ seems to be in their *circuit complexity*, which corresponds to our main point that circuit complexity considerations are crucial for the security of Goldreich’s PRG and OWF.

### 1.3 Conditional results for large polynomial stretch

Recall that the conjectured “hardness” of Goldreich’s PRG (i.e., Assumption 1) refers both to the regime of polynomial stretch and to the regime of quasi-polynomial stretch (as long as the locality is sufficiently large to support the corresponding stretch). Could it be that complexity-based attacks separate these two parameter regimes? That is, could the reason that our attacks from Section 1.2 work be that the stretch of the generator is super-polynomial?

As mentioned in Section 1.1 (and will be explained in Section 2), the underlying technical components in our complexity-based attacks are *unbalanced lossless expanders*  $[n] \times [m]$  whose neighbor functions have low circuit complexity, and *natural properties* against weak circuit classes. Our main results for the polynomial-stretch regime are of the following form: If lossless expanders  $[n] \times [n^{O(1)}]$  with constant degree and (specific) “very simple” neighbor functions exist, then exactly one of two cases holds:

1. Either the parameters of natural properties for certain well-studied weak circuit classes *cannot be improved*, even mildly; or
2. For a sufficiently large polynomial stretch, Goldreich’s generator is insecure when instantiated with a specific expander, *regardless of the predicate used*.

We now present two plausible assumptions on existence of suitable expanders, which are essentially improvements or extensions of existing explicit constructions. Conditioned on each assumption, we will contrast the security of Goldreich’s PRG with the possibility of extending natural proofs for some well-studied circuit class.

#### 1.3.1 Affine expanders and DNF-XOR circuits

As motivation for our first assumption, let us recall two well-known *explicit* constructions of unbalanced lossless expanders, which were given by Ta-Shma, Umans, and Zuckerman [30], and later on by Guruswami, Umans, and Vadhan [19]. We note that these two constructions are inherently different (the relevant construction from [30] is combinatorial, whereas the construction of [19] is algebraic), and yet in both constructions the neighbor function of the expander can be computed by *single layer of parity gates* (see [26, Sec. 5.1] for further details); we will call expanders with such a neighbor function *affine expanders*.

In the two foregoing affine expanders, the right-degree  $\ell$  is polylogarithmic, and it is an open problem to improve the degree to be constant, which matches the degree of a random construction. However, a random construction is not necessarily affine. Our first assumption is that there indeed exists an *affine expander* with *constant degree*:

► **Assumption 5** (expanders with an affine neighbor function; informal, see [26, Ass. 5.4]). *There exists  $\beta > 3$  such that for every constant  $k \in \mathbb{N}$  and sufficiently large  $n \in \mathbb{N}$ , there exists an  $(n^{.99}, 0.99)$ -expander  $G = ([n], [m = n^k], E)$  with right-degree  $\ell = \beta \cdot k$  whose neighbor function  $\Gamma_G : [m] \rightarrow ([n])^\ell$  can be computed by a single layer of parity gates.*

An unconditional proof of Assumption 5 will contrast the security of Goldreich’s PRG with the possibility of extending the known natural properties for DNF-XOR *circuits of exponential size*.<sup>5</sup> Specifically, known lower bounds for DNF-XOR circuits yield natural

---

<sup>5</sup> Recall that DNF-XOR circuits are depth-3 circuits that consist of a top OR gate, a middle layer of AND gates, and a bottom layer of parities above the inputs.



properties useful against such circuits of size up to  $2^{(1-o(1))\cdot n}$  (see [26, Sec. 5.1.2]).<sup>6</sup> Can these natural properties be extended to functions that are *approximated*, in the average-case sense, by DNF-XOR circuits of size  $2^{\epsilon n}$ , for some  $\epsilon > 0$ ? This is the natural property that we contrast with the security of Goldreich’s PRG:

► **Theorem 6** (is Goldreich’s generator insecure, or are natural properties for DNF-XOR circuits “non-extendable”?; informal statement). *Suppose that Assumption 5 holds. Then, exactly one of the following two options holds:*

1. *For all  $\epsilon > 0$ , there does not exist a natural property for the class of functions that can be approximated with success  $1/2 + o(1)$  by DNF-XOR circuits of size  $2^{\epsilon n}$ .*
2. *For a sufficiently large  $k \in \mathbb{N}$ , Goldreich’s generator is insecure with stretch  $m = n^k$  and locality  $\ell = \beta \cdot k$ , for some expander and regardless of the local predicate used.*

We stress that for any value of  $\beta > 3$  such that Assumption 5 holds, Theorem 6 follows with that value of  $\beta$ . Also note that Cohen and Shinkar [13] specifically conjectured that strong average-case lower bounds for DNF-XOR circuits of size  $2^{\Omega(n)}$  hold, and proved a similar statement for the related-yet-weaker model of parity decision trees. (Their proof for parity decision trees indeed yields a natural property; see [26, Prop. 5.13].)

### 1.3.2 $\mathcal{NC}^0$ expanders and weak $\mathcal{AC}_4^0$ circuits

To motivate our next assumption, recall the recent explicit construction of lossless expanders by Viola and Wigderson [33, Thm. 4] (which builds on the well-known construction of Capalbo *et al.* [11]). In this construction the neighbor function can be computed by an  $\mathcal{NC}^0$  circuit, but this construction is only for *balanced* expanders, rather than unbalanced ones. The following assumption is that such a construction is possible also for unbalanced expanders:

► **Assumption 7** (expanders with  $\mathcal{NC}^0$  neighbor functions; informal, see [26, Ass. 5.19]). *There exists  $\beta > 3$  such that for every constant  $k \in \mathbb{N}$  and sufficiently large  $n \in \mathbb{N}$ , there exists an  $(n^{.99}, 0.99)$ -expander  $G = ([n], [m = n^k], E)$  with right-degree  $\ell = \beta \cdot k$  such that the neighbor function  $\Gamma_G : [m] \rightarrow ([n])^\ell$  can be computed by an  $\mathcal{NC}^0$  circuit.*

An unconditional proof of Assumption 7 will immediately break Goldreich’s PRG in the polynomial-stretch regime by a complexity-based attack, when instantiated with a weak (but non-trivial) predicate class; see [26, Prop. 5.25]. But more importantly, such a proof will contrast the security of Goldreich’s PRG with the possibility of extending the known natural properties for the class of exponential-sized  $\mathcal{AC}^0$  circuits of depth four with constant bottom fan-in and top fan-in.

Since the precise trade-off between the parameters is a bit subtle, let us present the theorem in a simplified form (for a discussion of the more general setting, see [26, Sec. 5.2], and in particular [26, Sec. 5.2.3]). To do so, consider the (optimistic) possibility that in Assumption 7, there exists a single  $t$  such for any  $k \in \mathbb{N}$  the arity of the  $\mathcal{NC}^0$  circuit is  $t$  (i.e., each output bit of the circuit is a function of at most  $t$  input bits, where  $t$  does not depend on  $k$ ); as far as we are aware of, such a hypothesis is possible even with  $t = 1$ . Relying on Håstad’s switching lemma [21], for any  $c = O(1)$  there exists a natural property against depth-four circuits with top fan-in  $c$ , bottom fan-in  $t$ , and size  $2^{\epsilon \cdot (n/\log(c))}$  for a tiny universal

<sup>6</sup> Some of these natural properties actually run in slightly super-polynomial time, rather than in strictly polynomial time, but this issue is not crucial for our purpose of breaking Goldreich’s PRG.

$\epsilon > 0$  (see [26, Cor. 5.24]). In the following theorem, the security of Goldreich’s PRG is contrasted with the possibility of extending these natural properties to work against such circuits of size  $2^{\beta \cdot (n/\log(c))}$  where  $\beta > 3$ .

► **Theorem 8** (is Goldreich’s generator insecure, or are natural properties for very restricted  $\mathcal{AC}^0$  circuits “non-extendable”?; informal statement). *Suppose that Assumption 7 holds and that for any  $k \in \mathbb{N}$ , the arity of the  $\mathcal{NC}^0$  circuit equals  $t = O(1)$ . Then, exactly one of the following two options holds:*

1. *For any  $c \in \mathbb{N}$ , there does not exist a natural property for depth-four  $\mathcal{AC}^0$  circuits with top fan-in  $c$  and bottom fan-in  $t$  and size  $O(2^{\beta \cdot (n/\log(c))})$ .*
2. *For a sufficiently large  $k \in \mathbb{N}$ , Goldreich’s generator is insecure with stretch  $m = n^k$  and predicate locality  $\ell = \beta \cdot k$ , for some expander and regardless of the predicate used.*

Recall that Assumption 7 is parametrized by  $\beta$  and by the arity of the  $\mathcal{NC}^0$  circuit; we stress that for *any* values of  $\beta$  and  $t$  such that Assumption 7 holds, we get a corresponding “win-win” theorem such as Theorem 8 (for further details see [26, Sec. 5.2]). We also stress that both the natural properties that we can unconditionally prove and the natural properties referred to in Theorem 8 are for circuits of exponential size  $2^{\Theta(n/\log(c))}$ , and the difference is in the universal constant hidden in the  $\Theta$ -notation.

As mentioned in Section 1.1, the *explicit* construction of highly unbalanced lossless expanders is a long-standing open problem, regardless of the circuit complexity of their neighbor function (see, e.g., [11], [32, Prob. 5.36 & 6.35], and [34, Chap. 8.7]). Assumptions 5 and 7, however, *do not concern explicit constructions* of expanders, but only assume their *existence*; in particular, the circuit family for the neighbor function of the graph may be *non-uniform*. (This is indeed the case for our construction of expanders in the quasi-polynomial stretch regime.)

## 2 Overviews of the proofs

### 2.1 The general form of attack

A natural property for a class  $\mathcal{F}$  of functions is a deterministic polynomial-time algorithm that rejects all truth-tables of functions from  $\mathcal{F}$ , but accepts the truth-tables of almost all functions.<sup>7</sup> Indeed, a natural property for  $\mathcal{F}$  exists only if almost all functions are *not* in  $\mathcal{F}$ . We will show how to use natural properties to break Goldreich’s pseudorandom generator.

The key step in our proofs is to show, for every fixed  $x \in \{0, 1\}^n$ , that  $\text{prg}(x)$  is the truth-table of a function from some class  $\mathcal{F}$  of “simple” functions (e.g.,  $\text{prg}(x)$  is the truth-table of a small constant-depth circuit). When we are able to show this, it follows that a **natural property** for  $\mathcal{F}$  can distinguish the outputs of the PRG from uniformly-chosen random strings: This is because the natural property rejects any string in the output-set of the PRG (which is the truth-table of a function in  $\mathcal{F}$ ), but accepts a random string, with high probability. (The general idea of using natural properties to break PRGs in this manner goes back to the original work of [28].)

Recall that Goldreich’s PRG (i.e., the function  $\text{prg}$ ) is *always* a very “simple” function, since each output bit depends on a few (i.e.,  $\ell \ll n$ ) input bits. However, in order for our idea to work, we need that a *different* function (i.e., not the function  $\text{prg}$ ) will be simple:

<sup>7</sup> Throughout the paper, we identify a natural property with the “constructive” algorithm that recognizes the property (see [26, Def. 3.8]).



Specifically, for every *fixed* input  $x$ , we want that the function  $g_x : \{0, 1\}^{\log(m)} \rightarrow \{0, 1\}$  such that  $g_x(i) = \text{prg}(x)_i$  will be “simple”. That is, for a *fixed* “seed”  $x$  for the PRG, the function  $g_x$  gets as input an index  $i$  of an output bit, and computes the  $i^{\text{th}}$  output bit of  $\text{prg}(x)$  as a function of  $i$ . Intuitively, given  $i \in [m]$ , the function  $g_x$  needs to compute three different objects, successively:

- The neighbors  $\Gamma_G(i)$  of the vertex  $i \in [m]$  in  $G$ .
- The projections of the (fixed) string  $x$  on locations  $\Gamma_G(i)$ .
- The output of the predicate  $P$  on  $x|_{\Gamma_G(i)}$ .

The proofs of our main theorems consist of showing instantiations of Goldreich’s generator (i.e., choices for an expander and a predicate) such that  $g_x$  is a function from a class against which we can construct natural properties.

An alternative view of the construction of  $g_x$  above is as giving rise to a collection of *pseudorandom functions* (PRFs)  $\{g_x : \{0, 1\}^{\log(m)} \rightarrow \{0, 1\}\}_{x \in \{0, 1\}^n}$  that are based on (an instantiation of) Goldreich’s PRG. In fact, the construction of  $g_x$  is technically reminiscent of constructions of PRFs that are based on Goldreich’s PRG by Applebaum and Raykov [8]. However, the crucial point is that our transformation of Goldreich’s PRG to a PRF incurs very little complexity overhead; in particular, the circuit complexity of  $g_x$  is essentially determined by the circuit complexity of the expander’s neighbor function and of the predicate. For further discussion see Section 2.4.

## 2.2 The setting of quasi-polynomial stretch

The proof of Theorem 2 consists of showing that for a suitable expander  $G$ , and for any predicate  $P$  computable by an  $\mathcal{AC}^0[\oplus]$  circuit of sufficiently small sub-exponential size, the function  $g_x$  can be computed by an  $\mathcal{AC}^0[\oplus]$  circuit of sufficiently small sub-exponential size. Natural properties for such circuits, based on the lower bounds by Razborov and Smolensky [27, 29], are well-known (see, e.g., [28, 12]).

To describe the instantiations and the construction of an  $\mathcal{AC}^0[\oplus]$  circuit for  $g_x$ , let  $n \in \mathbb{N}$ , and let  $m = 2^{(\log(n))^k}$ , for a sufficiently large  $k$ . The first technical component that we need is an expander graph  $G$  such that the function  $i \mapsto \Gamma_G(i)$  can be computed by a sub-exponential sized  $\mathcal{AC}^0[\oplus]$  circuit. We show that there exists such a graph, with essentially optimal parameters:

► **Theorem 9** (strongly-explicit lossless expander in  $\mathcal{AC}^0[p]$ ; see [26], Thm. 4.5). *There exists a universal constant  $d_G \in \mathbb{N}$  such that the following holds. For any  $k \in \mathbb{N}$  and sufficiently large  $n$  and  $m = 2^{(\log(n))^k}$ , there exists a  $(n^{0.99}, 0.99)$ -expander  $G = ([n], [m], E)$  of right-degree  $\ell = O(\log(m)/\log(n))$ , and an  $\mathcal{AC}^0[\oplus]$  circuit  $C_G : \{0, 1\}^{\log(m)} \rightarrow \{0, 1\}^{\ell \cdot \log(n)}$  of depth  $d_G$  and size  $\text{poly}(n)$  such that for every  $i \in [m]$  it holds that  $C_G(i)$  outputs the list of  $\ell$  neighbors of  $i$  in  $G$ .*

We stress that the depth  $d_G$  of the circuit in Theorem 9 does not depend on the relation between  $m$  and  $n$ , which is what will allow us to have a natural property for the circuit  $C_G$ . Specifically, recall that we have natural properties against  $\mathcal{AC}^0[\oplus]$  circuits of depth  $d_G$  over  $\ell_m = \log(m)$  input bits of sub-exponential size  $2^{\Omega(\ell_m^{1/2d_G})}$ . The size of  $C_G$  is  $\text{poly}(n)$ , and thus if we take  $m = 2^{(\log(n))^k}$ , for a sufficiently large  $k$ , then the size of  $C_G$  is a sufficiently small sub-exponent in its input length  $\log(m)$ .

In high-level, our construction of the expander in Theorem 9 is as follows. Our starting point is the well-known fact that a random graph is, with high probability, a good lossless bipartite expander (see, e.g., [26, Thm. 3.2]). The first step is to construct an *efficient*

*test* that gets as input a string  $G \in \{0, 1\}^{m'}$ , where  $m' = m \cdot \ell \cdot \log(n)$ , considers  $G$  as the incidence-list of a graph, and decides whether or not  $G$  is an  $(n^{.99}, .99)$ -expander. We show that such a test can be implemented by a CNF of size  $2^n$  (see [26, Clm. 4.2]). Hence, a pseudorandom generator for CNFs of size  $2^n$  outputs, with high probability, a good expander. Specifically, we will use the pseudorandom generator of Nisan [24], which has seed length  $\text{poly}(n)$ . Thus, for some fixed “good” seed  $s$ , the output  $NW(s) \in \{0, 1\}^{m'}$  of the generator on  $s$  is an  $(n^{.99}, .99)$ -expander.

Our next step is to show that the expander represented by  $NW(s)$  has neighbor functions that can be computed by an  $\mathcal{AC}^0[\oplus]$  circuit. In fact, we will show that there exists a circuit that gets as input the index  $i \in \{0, 1\}^{\log(m')}$  of a bit in  $NW(s)$  and outputs  $NW(s)_i$ . To do so we can rely, for instance, on the recent work of Carmosino *et al.* [12], who showed that Nisan’s generator can be made “strongly-explicit”: That is, there exists an  $\mathcal{AC}^0[\oplus]$  circuit of polynomial size that gets as input a seed  $z$  and an index  $i$  of an output bit, and computes the  $i^{\text{th}}$  output bit of the generator on seed  $z$ .<sup>8</sup> By “hard-wiring” a “good” seed  $s$  into the latter circuit, we obtain an  $\mathcal{AC}^0[\oplus]$  circuit of size  $\text{poly}(n)$  that computes the output bits of the expander  $NW(s)$ . Indeed, a crucial point is that we did not algorithmically look for a good seed  $s$ , but rather non-uniformly fixed a “good” seed and “hard-wired” it into the circuit.

Given this expander construction,  $g_x$  can compute  $i \mapsto \Gamma_G(i)$  in sub-exponential size, and we now need  $g_x$  to compute the projections of  $x$  on locations  $\Gamma_G(i)$ . To do so we simply “hard-wire” the entire string  $x$  into  $g_x$ . Specifically, after computing the function  $i \mapsto \Gamma_G(i)$ , the circuit now has the  $\ell \cdot \log(n)$  bits of  $\Gamma_G(i)$ ; it then uses  $\ell$  depth-two formulas, each over  $\log(n)$  bits and of size  $n$ , to compute the mapping  $\Gamma_G(i) \mapsto x|_{\Gamma_G(i)}$  by brute-force. This increases the size of the circuit for  $g_x$  by  $\ell \cdot n < n^2$  gates, which is minor compared to the size  $\text{poly}(n)$  of  $C_G$  from Theorem 9.

Finally, the circuit  $g_x$  has now computed the  $\ell$  bits corresponding to  $x|_{\Gamma_G(i)}$ , and needs to compute the predicate  $P : \{0, 1\}^\ell \rightarrow \{0, 1\}$  on these bits. To get the circuit to be of sufficiently small sub-exponential size, we require that the predicate can be computed by a sufficiently small sub-exponential sized  $\mathcal{AC}^0[\oplus]$  circuit. Specifically, we want that for some  $d_P$ , the predicate  $P$  can be computed by an  $\mathcal{AC}^0[\oplus]$  circuit of depth  $d_P$  and size  $2^{\ell^\epsilon}$ , for a sufficiently small  $\epsilon < 1/2(d_G + d_P + 2)$ . We thus obtain a circuit for  $g_x$  of depth  $d = d_G + d_P + 2$  and of size  $O(2^{\ell^\epsilon}) < 2^{\log(m)^{1/2d}}$ ,<sup>9</sup> which is sufficiently small such that we have natural properties against it (for a formal statement of the parameters of this well-known natural property, proved by [28, 12], see e.g. [26, Thm. 3.9]).

### 2.3 The setting of large polynomial stretch

Why are the results in Section 2.2 applicable only to the setting of quasi-polynomial stretch? The main bottleneck is the expander construction in Theorem 9, which is an  $\mathcal{AC}^0[\oplus]$  circuit. Specifically, since we only know of natural properties against  $\mathcal{AC}^0[\oplus]$  circuits of at most sub-exponential size, and since the circuit that we obtain is of size at least  $n$  (because we hard-wire  $x \in \{0, 1\}^n$  to the circuit), we were forced to take  $m = n^{\text{poly} \log(n)}$  such that  $n$  will be a small sub-exponential function of  $\log(m)$ .

In this section we circumvent this obstacle by using the hypothesized existence of expanders whose neighbor functions have “extremely simple” circuits. For simplicity, in the current high-level overview we present the attacks that are based on the existence of an expander

<sup>8</sup> A similar observation has appeared in other works, such as in [28, Thm. 4.2].

<sup>9</sup> For this calculation we assumed that  $2^{\ell^\epsilon}$  dominates the size of the circuit (since the size of  $C_G$  is already sufficiently small); and we used the fact that  $\ell = O(\log(m)/\log(n)) < \log(m)$ , and that  $\epsilon < 1/2d$  is sufficiently small).

as in Assumption 5; that is, a lossless expander  $G = ([n], [m = n^k], E)$  of right-degree  $\ell = O(k)$  whose neighbor function is an *affine function* (i.e., each output bit is a parity of input bits). The ideas that underlie the attacks that are based on expanders whose neighbor function is an  $\mathcal{NC}^0$  circuit (as in Assumption 7) are similar, yet require a slightly more subtle parametrization (see [26, Sec. 5.2]).

Consider an instantiation of Goldreich’s predicate with expander  $G$  as above and with a predicate  $P : \{0, 1\}^\ell \rightarrow \{0, 1\}$  that can be computed by a CNF of size  $2^{\delta \cdot \ell}$ , where  $\delta$  can be an arbitrarily large constant compared to  $k$  (or even  $\delta = 1$ , which allows for *any* predicate). In this case, for any  $x \in \{0, 1\}^n$ , the output  $\text{prg}(x)$  of the generator on  $x$  is a truth-table of a function  $g_x$  over an input  $i \in \{0, 1\}^{\log(m)}$  that can be computed as follows. One layer of *parity gates* maps  $i \in [m]$  to  $\Gamma_G(i) \in \{0, 1\}^{\ell \cdot \log(n)}$  (this uses our assumption about the expander). Then,  $\ell$  copies of a DNF over  $\log(n)$  bits and of size  $n$  map the names of the  $\ell$  vertices to  $x|_{\Gamma_G(i)} \in \{0, 1\}^\ell$ , i.e., we project the bits of  $x$  that feed the predicate  $P$  (this DNF is essentially a “hard-wiring” of  $x$  into  $g_x$ ). Finally, the CNF that computes  $P$  of size  $2^{\delta \cdot \ell}$  maps  $x|_{\Gamma_G(i)}$  to the value  $P(x|_{\Gamma_G(i)})$ . After collapsing a layer that connects the top CNF and the DNFs, we obtain an AND-OR-AND-XOR circuit  $g_x$  over  $\ell_m = \log(m)$  input bits of size  $O(\ell \cdot \log(n) + \ell \cdot n + 2^{\ell \cdot \delta}) = O(2^{\ell_m/k})$  with top fan-in  $2^{\delta \cdot \ell} = 2^{O(\delta \cdot k)}$ .

When  $\delta > 0$  is sufficiently small, we are able to unconditionally construct a natural property against circuits as above. However, the main point (i.e., Theorem 6) comes when considering the case  $\delta = 1$ ; that is, *any* predicate  $P : \{0, 1\}^\ell \rightarrow \{0, 1\}$ . In this case, we first use the discriminator lemma of [20] to deduce that  $g_x$  can be  $(1/2 + 1/2^{O(k)})$ -approximated by a DNF-XOR circuit of size  $O(2^{\ell_m/k})$ . Now (still under Assumption 5), exactly one of two options holds. The first option is that there exists a natural property for functions on  $\ell_m$  input bits that can be  $(1/2 + o(1))$ -approximated by DNF-XOR circuits of size  $2^{\Omega(\ell_m)}$ ; in this case, by taking  $k$  sufficiently large so that  $2^{\ell_m/k}$  is sufficiently small, the natural property breaks the generator. The other option is that no such natural property exists, despite the fact that natural properties exist both for functions computed (in the worst-case) by DNF-XOR circuits of size  $2^{(1-o(1)) \cdot \ell_m}$ , and for functions approximated (even weakly) by parity decision trees of such size. This completes the sketch of the proof of Theorem 6.

## 2.4 The connection to expander-based pseudorandom functions

As mentioned in Section 2.1, our construction of the function  $g_x : \{0, 1\}^{\log(m)} \rightarrow \{0, 1\}$  (i.e.,  $g_x(i) = P(x|_{\Gamma_G(i)})$ ) can be viewed as a construction of a collection of *pseudorandom functions* (PRFs)  $\{g_x : \{0, 1\}^{\log(m)} \rightarrow \{0, 1\}\}_{x \in \{0, 1\}^n}$  based on (an instantiation of) Goldreich’s PRG. The crucial point in our transformation of Goldreich’s PRG to a PRF is that the resulting PRF can have very low circuit complexity, depending essentially only on the complexity of the expander’s neighbor function and of the predicate. In contrast, previously-known transformations of Goldreich’s PRG to a PRF incur a significant overhead. Specifically, the transformation of Goldreich, Goldwasser, and Micali [18] yields a circuit with super-constant depth; whereas the constructions of Applebaum and Raykov [8] either yield only a *weak* PRF (which is not broken by natural properties, in general) or require complicated computations, which they implement using majority gates (i.e., the resulting function is in the class  $\mathcal{TC}^0$ , for which natural properties are neither known nor conjectured to exist).

Nevertheless, as pointed out by Applebaum,<sup>10</sup> a transformation of Goldreich’s PRG to a *weak* PRF from [8] can be used to break the PRG when it is instantiated with a *random graph* and with a predicate with sufficiently low circuit complexity; this attack uses algorithms for

<sup>10</sup>Personal communication.

learning from random examples (instead of natural properties). Specifically, assume that Goldreich’s PRG is secure when instantiated with a random graph  $[n] \times [m]$  of right-degree  $\ell$  and a predicate  $P : \{0, 1\}^\ell \rightarrow \{0, 1\}$ . Using the argument that appears in [8, Sec. 1.2.1] it follows that the function  $g_x : \{0, 1\}^{\ell \cdot \log(n)} \rightarrow \{0, 1\}$  that considers its input as a set  $S$  of  $\ell$  vertices in  $[n]$ , and outputs  $g_x(S) = P(x|_S)$ , is a weak PRF against adversaries that make  $m$  (uniformly-chosen) queries. The complexity of  $g_x$  is essentially determined by the complexity of the predicate  $P$ .<sup>11</sup> Thus, if the latter is sufficiently small such that there exists an algorithm for learning  $g_x$  from  $m - 1$  random examples, then  $g_x$  cannot be a weak PRF for adversaries that make  $m$  queries (since such an adversary can use the learning algorithm to predict the  $m^{\text{th}}$  evaluation of the function at a random point, using the first  $m - 1$  evaluations at random points). This contradicts the hypothesis that Goldreich’s PRG is secure when instantiated with the predicate  $P$  and a random graph  $[n] \times [m]$ .

Loosely speaking, the argument above implies that Goldreich’s PRG is not secure when the stretch is quasipolynomial (and the locality is polylogarithmic and sufficiently large), the graph is random, and the predicate is computable by an  $\mathcal{AC}^0$  circuit of sufficiently small sub-exponential size; this relies on the learning algorithm of Linial, Mansour, and Nisan [22].<sup>12</sup> However, the latter class of predicates is much weaker than the class of predicates to which our main unconditional result applies (i.e., than the class of  $\mathcal{AC}^0[\oplus]$  circuits of sufficiently small sub-exponential size, from Theorem 2). For example, such predicates have “low” resilience  $o(\ell)$ , because the Fourier weight of depth- $d$   $\mathcal{AC}^0$  circuits over  $\ell$  bits of size  $2^{\ell^\epsilon}$  is .01-concentrated on sets of size at most  $O(\ell^{\epsilon \cdot (d-1)}) = o(\ell)$  (see [22, 31]); therefore, such predicates do not withstand the attacks from [7]. Finally, recall that it is currently an open problem to understand the learnability of  $\mathcal{AC}^0[\oplus]$  circuits from random examples.

---

## References

- 1 Michael Alekhnovich. More on average case vs approximation complexity. *Computational Complexity*, 20(4):755–786, 2011.
- 2 Benny Applebaum. *Cryptography in Constant Parallel Time*. Information Security and Cryptography. Springer, 2014.
- 3 Benny Applebaum. Cryptographic Hardness of Random Local Functions. *Computational Complexity*, 25(3):667–722, 2016.
- 4 Benny Applebaum, Boaz Barak, and Avi Wigderson. Public-key cryptography from different assumptions. In *Proc. 42nd Annual ACM Symposium on Theory of Computing (STOC)*, pages 171–180. ACM, 2010.
- 5 Benny Applebaum, Andrej Bogdanov, and Alon Rosen. A dichotomy for local small-bias generators. *Journal of Cryptology*, 29(3):577–596, 2016.
- 6 Benny Applebaum, Yuval Ishai, and Eyal Kushilevitz. Cryptography in  $\text{NC}^0$ . *SIAM Journal of Computing*, 36(4):845–888, 2006.
- 7 Benny Applebaum and Shachar Lovett. Algebraic attacks against random local functions and their countermeasures. *SIAM Journal of Computing*, 47:52–79, 2018.

---

<sup>11</sup> More specifically, the circuit  $g_x$  can be implemented by  $2 \cdot \ell$  depth-two formulas of size  $O(n)$  to compute the mapping  $S \mapsto x|_S$ , and then a circuit for  $P$  to compute  $P(x|_S)$ .

<sup>12</sup> Specifically, let  $n \in \mathbb{N}$ , and let  $\ell = \log^k(n)$  for some  $k \in \mathbb{N}$ . Assume that the predicate  $P : \{0, 1\}^\ell \rightarrow \{0, 1\}$  is computable by an  $\mathcal{AC}^0$  circuit of depth  $d \in \mathbb{N}$  and size  $s = 2^{\ell^\epsilon}$ , where  $\epsilon \leq 1/(d+1)$ . Then,  $g_x$  can be implemented by a circuit of size  $s' = O(\ell \cdot n + s)$  and depth  $d+1$ . The algorithm of [22] learns  $g_x$  with error  $1/s'$  from  $m = n^{O(\log(s'))^d} = o(n^\ell)$  random examples in time  $\text{poly}(m)$ , where the last bound on  $m$  is since  $\epsilon \leq 1/(d+1)$ . Therefore, Goldreich’s PRG is not secure when the locality is  $\ell = \log^k(n)$ , the stretch is  $m = o(n^\ell)$ , and the  $\ell$ -bit predicate is an  $\mathcal{AC}^0$  circuit of depth  $d$  and size  $2^{\ell^\epsilon}$ .

- 8 Benny Applebaum and Pavel Raykov. Fast pseudorandom functions based on expander graphs. In *Theory of cryptography. Part I*, volume 9985 of *Lecture Notes in Comput. Sci.*, pages 27–56. Springer, Berlin, 2016.
- 9 Andrej Bogdanov and Youming Qiao. On the security of Goldreich’s one-way function. *Computational Complexity*, 21(1):83–127, 2012.
- 10 Andrej Bogdanov and Alon Rosen. Input locality and hardness amplification. *Journal of Cryptology*, 26(1):144–171, 2013.
- 11 Michael Capalbo, Omer Reingold, Salil Vadhan, and Avi Wigderson. Randomness Conductors and Constant-degree Lossless Expanders. In *Proc. 34th Annual ACM Symposium on Theory of Computing (STOC)*, pages 659–668, 2002.
- 12 Marco L. Carmosino, Russell Impagliazzo, Valentine Kabanets, and Antonina Kolokolova. Learning algorithms from natural proofs. In *Proc. 31st Annual IEEE Conference on Computational Complexity (CCC)*, page 10 (24), 2016.
- 13 Gil Cohen and Igor Shinkar. The complexity of DNF of parities. In *Proc. 7th Annual Innovations in Theoretical Computer Science Conference (ITCS)*, pages 47–58. ACM, 2016.
- 14 James Cook, Omid Etesami, Rachel Miller, and Luca Trevisan. On the one-way function candidate proposed by Goldreich. *ACM Transactions of Computation Theory*, 6(3):Art. 14, 35, 2014.
- 15 Vitaly Feldman, Will Perkins, and Santosh Vempala. On the Complexity of Random Satisfiability Problems with Planted Solutions. In *Proc. 47th Annual ACM Symposium on Theory of Computing (STOC)*, pages 77–86, 2015.
- 16 Oded Goldreich. Candidate One-Way Functions Based on Expander Graphs. *Electronic Colloquium on Computational Complexity: ECCC*, 7:90, 2000.
- 17 Oded Goldreich. Candidate one-way functions based on expander graphs. In *Studies in complexity and cryptography*, volume 6650 of *Lecture Notes in Computer Science*, pages 76–87. Springer, Heidelberg, 2011.
- 18 Oded Goldreich, Shafi Goldwasser, and Silvio Micali. How to construct random functions. *Journal of the ACM*, 33(4):792–807, 1986.
- 19 Venkatesan Guruswami, Christopher Umans, and Salil Vadhan. Unbalanced expanders and randomness extractors from Parvaresh-Vardy codes. *Journal of the ACM*, 56(4):Art. 20, 34, 2009.
- 20 András Hajnal, Wolfgang Maass, Pavel Pudlák, Mária Szegedy, and György Turán. Threshold circuits of bounded depth. *Journal of Computer and System Sciences*, 46(2):129–154, 1993.
- 21 Johan Håstad. *Computational Limitations of Small-depth Circuits*. MIT Press, 1987.
- 22 Nathan Linial, Yishay Mansour, and Noam Nisan. Constant depth circuits, Fourier transform, and learnability. *Journal of the Association for Computing Machinery*, 40(3):607–620, 1993.
- 23 Elchanan Mossel, Amir Shpilka, and Luca Trevisan. On  $\epsilon$ -biased generators in  $NC^0$ . *Random Structures & Algorithms*, 29(1):56–81, 2006.
- 24 Noam Nisan. Pseudorandom bits for constant depth circuits. *Combinatorica*, 11(1):63–70, 1991.
- 25 Ryan O’Donnell and David Witmer. Goldreich’s PRG: evidence for near-optimal polynomial stretch. In *Proc. 29th Annual IEEE Conference on Computational Complexity (CCC)*, pages 1–12. IEEE, 2014.
- 26 Igor Carboni Oliveira, Rahul Santhanam, and Roei Tell. Expander-Based Cryptography Meets Natural Proofs. *Electronic Colloquium on Computational Complexity: ECCC*, 25:159, 2018. URL: <https://eccc.weizmann.ac.il/report/2018/159/>.

## 18:14 Expander-Based Cryptography Meets Natural Proofs

- 27 Alexander A. Razborov. Lower bounds on the size of constant-depth networks over a complete basis with logical addition. *Mathematical Notes of the Academy of Science of the USSR*, 41(4):333–338, 1987.
- 28 Alexander A. Razborov and Steven Rudich. Natural proofs. *Journal of Computer and System Sciences*, 55(1, part 1):24–35, 1997.
- 29 Roman Smolensky. Algebraic Methods in the Theory of Lower Bounds for Boolean Circuit Complexity. In *Proc. 19th Annual ACM Symposium on Theory of Computing (STOC)*, pages 77–82, 1987.
- 30 Amnon Ta-Shma, Christopher Umans, and David Zuckerman. Lossless condensers, unbalanced expanders, and extractors. *Combinatorica*, 27(2):213–240, 2007.
- 31 Avishay Tal. Tight Bounds on the Fourier Spectrum of AC0. In *Proc. 32nd Annual IEEE Conference on Computational Complexity (CCC)*, pages 15:1–15:31, 2017.
- 32 Salil P. Vadhan. *Pseudorandomness*. Foundations and Trends in Theoretical Computer Science. Now Publishers, 2012.
- 33 Emanuele Viola and Avi Wigderson. Local Expanders. *Computational Complexity*, 2017.
- 34 Avi Wigderson. *Mathematics and Computation (book draft)*, August 26, 2018. Accessed at <https://www.math.ias.edu/avi/book>, August 26, 2018.



# A Note on the Quantum Query Complexity of Permutation Symmetric Functions

André Chailloux

Inria de Paris, EPI SECRET, Paris, France

andre.chailloux@inria.fr

---

## Abstract

It is known since the work of [1] that for any permutation symmetric function  $f$ , the quantum query complexity is at most polynomially smaller than the classical randomized query complexity, more precisely that  $R(f) = \tilde{O}(Q^7(f))$ . In this paper, we improve this result and show that  $R(f) = O(Q^3(f))$  for a more general class of symmetric functions. Our proof is constructive and relies largely on the quantum hardness of distinguishing a random permutation from a random function with small range from Zhandry [11].

**2012 ACM Subject Classification** Theory of computation → Quantum query complexity

**Keywords and phrases** quantum query complexity, permutation symmetric functions

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2019.19

**Acknowledgements** The author wishes to thank Anthony Leverrier and anonymous referees for helpful comments on this paper.

## 1 Introduction

The black box model has been a very fruitful model for understanding the possibilities and limitations of quantum algorithms. In this model, we can prove some exponential speedups for quantum algorithms, which is notoriously hard to do in standard complexity theory. Famous examples are the Deutsch-Josza problem [7] and Simon's problem [10]. There has been a great line of work to understand quantum query complexity, which developed some of the most advanced algorithms techniques. Even Shor's algorithm [9] for factoring fundamentally relies on a black box algorithm for period finding.

We describe here the query complexity model in a nutshell. The idea is that we have to compute  $f(x_1, \dots, x_n)$  where each  $x_i \in [M]$  can be accessed via a query. We consider decision problems meaning that  $f : S \rightarrow \{0, 1\}$  with  $S \subseteq [M]^n$ . In this paper, we will consider inputs  $x \in [M]^n$  equivalently as functions from  $[n] \rightarrow [M]$ . We are not interested in the running time of our algorithm but only want to minimize the number of queries to  $x$ , which in the quantum setting consists of applying the unitary  $\mathcal{O}_x : |i\rangle|j\rangle \rightarrow |i\rangle|j + x_i\rangle$ .  $D(f)$ ,  $R(f)$  and  $Q(f)$  represent the minimal amount of queries to compute  $f$  with probability greater than  $2/3$  (or  $= 1$  for the case of  $D(f)$ ) using respectively a deterministic algorithm with classical queries, a randomized algorithm with classical queries and a quantum algorithm with quantum queries.

As we said before, the query complexity is great for designing new quantum algorithms. It is also very useful for providing black box limitations for quantum algorithms. There are some cases in particular where we can prove that the quantum query complexity of  $f$  is at most polynomially smaller than classical (deterministic or randomized) query complexity. For example:



© André Chailloux;

licensed under Creative Commons License CC-BY

10th Innovations in Theoretical Computer Science (ITCS 2019).

Editor: Avrim Blum; Article No. 19; pp. 19:1–19:7

Leibniz International Proceedings in Informatics



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



- for specific functions such as search [6] or element distinctness (ED) [2, 8, 4], we have respectively  $Q(\text{Search}) = \Theta(n^{1/2})$ ,  $D(\text{Search}) = \Theta(n)$  and  $Q(\text{ED}) = \Theta(n^{2/3})$ ,  $D(\text{ED}) = \Theta(n)$ .
- For any total function  $f$  i.e. when its domain  $S = [M]^n$ , Beals *et al.* [5] proved using the polynomial method that  $D(f) \leq O(Q^6(f))$ .

Another case of interest where we can lower bound the quantum query complexity is the case of permutation symmetric functions. There are several ways of defining such functions and we will be interested in the following definitions for a function  $f : S \rightarrow \{0, 1\}$  with  $S \subseteq [M]^n$ .

► **Definition 1.**

- $f$  permutation symmetric of the first type iff.  $\forall \pi \in S_n, f(x) = f(x \circ \pi)$ .
- $f$  is permutation symmetric of the second type iff.  $\forall \pi \in S_n, \forall \sigma \in S_M, f(x) = f(\sigma \circ x \circ \pi)$ .

where  $S_n$  (resp.  $S_M$ ) represents the set of permutations on  $[n]$  (resp.  $[M]$ ) and  $\circ$  is the usual function composition.

Here, recall that we consider strings  $x \in [M]^n$  as functions from  $[n] \rightarrow [M]$ . Notice also that this definition implies that  $S$  is stable by permutation, meaning that  $x \in S \Leftrightarrow \forall \pi \in S_n, x \circ \pi \in S$ . We already know from the work of Aaronson and Ambainis the following result:

► **Theorem 2** ([1]). *For any permutation symmetric function  $f$  of the second type (Definition 1),  $R(f) \leq \tilde{O}(Q^7(f))$ .*

In a recent survey on quantum query complexity and quantum algorithms [3], Ambainis writes:

“It has been conjectured since about 2000 that a similar result also holds for  $f$  with a symmetry of the first type.”

**Contribution**

The contribution of this paper is to prove the above conjecture. We show the following:

► **Theorem 3.** *For any permutation symmetric function  $f$  of the first type,  $R(f) \leq O(Q^3(f))$ .*

This result not only generalizes the result for a more general class of permutation symmetric function, but also improves the exponent from 7 to 3. In the case where  $M = 2$ , this result was already known [1] with an exponent of 2, which is tight from Grover’s algorithm.

The proof technique is arguably simple, constructive and relies primarily on the quantum hardness of distinguishing a random permutation from a random function with small range from Zhandry [11]. We start from a permutation symmetric function  $f$ . At high level, the proof goes as follows:

- We start from an algorithm  $\mathcal{A}$  that outputs  $f(x)$  for all  $x$  with high (constant) probability. Let  $q$  the number of quantum queries to  $\mathcal{O}_x$  performed by  $\mathcal{A}$ .
- Instead of running  $\mathcal{A}$  on input  $x$ , we choose a random function  $C : [n] \rightarrow [n]$  with a range of small size  $r$  (from a distribution specified later in the paper) and apply the algorithm  $\mathcal{A}$  where we replace calls to  $\mathcal{O}_x$  with calls to  $\mathcal{O}_{x \circ C}$ . We note that there is a simple procedure to compute  $\mathcal{O}_{x \circ C}$  from  $\mathcal{O}_x$  and  $\mathcal{O}_C$ .

- If we take  $r = \Theta(q^3)$ , we can use Zhandry's lower bound, we show that for each  $x$ , the output will be close to the output of the algorithm  $\mathcal{A}$  where we replace calls to  $\mathcal{O}_{x \circ C}$  with calls to  $\mathcal{O}_{x \circ \pi}$  for a random permutation  $\pi$ . Using the fact that  $f$  is permutation symmetric, the latter algorithm will output with high probability  $f(x \circ \pi) = f(x)$ . In other words, if the algorithm  $\mathcal{A}$  that calls  $\mathcal{O}_{x \circ C}$  wouldn't output  $f(x)$  for a random  $C$  and a fixed  $x$  then we would find a distinguisher between a random  $C$  and a random permutation  $\pi$ , which is hard from Zhandry's lower bound.
- The above tells us that applying  $\mathcal{A}$  where we replace calls to  $\mathcal{O}_x$  with calls to  $\mathcal{O}_{x \circ C}$  gives us output  $f(x)$  with high probability. Knowing  $C$ , we can construct the whole string  $x \circ C$  by querying  $x$  on inputs  $i \in \text{Im}(C)$  which can be done with  $|\text{Im}(C)| \leq r$  classical queries which allows us to construct the unitary  $\mathcal{O}_{x \circ C}$ . This means we can emulate  $\mathcal{A}$  on input  $x \circ C$  with  $r$  classical queries to  $x$  and this gives us  $f(x)$  with high probability.

After presenting a few notations, we dive directly into the proof of our theorem.

## 2 Preliminaries

### 2.1 Notations

For any function  $f$ , let  $\text{Im}(f)$  be its range (or image).

#### Query algorithms

A query algorithm  $\mathcal{A}^\mathcal{O}$  is described by an algorithm that calls another function  $\mathcal{O}$  in a black box fashion. We will never be interested in the running time or the size of  $\mathcal{A}$  but only in the number of calls, or queries, to  $\mathcal{O}$ . We will consider both the cases where the algorithm  $\mathcal{A}^\mathcal{O}$  is classical and quantum. In the latter  $\mathcal{O}$  will be a quantum unitary. In both cases, we only consider algorithms that output a single bit.

#### Oracles

We use oracles to perform black box queries to a function. For any function  $g$ ,  $\mathcal{O}_g^{\text{Classical}}$  is a black box that on input  $i$  outputs  $g(i)$  while  $\mathcal{O}_g$  (without any superscript) is the quantum unitary satisfying

$$\mathcal{O}_g : |i\rangle|j\rangle \rightarrow |i\rangle|j + g(i)\rangle.$$

#### Query complexity

Fix a function  $f : S \rightarrow \{0, 1\}$  where  $S \subseteq [M]^n$ .

► **Definition 4.** The randomized query complexity  $R(f)$  of  $f$  is the smallest integer  $q$  such that there exists a classical randomized algorithm  $\mathcal{A}^\mathcal{O}$  performing  $q$  queries to  $\mathcal{O}$  satisfying:

$$\forall x \in S, \Pr[\mathcal{A}_x^{\mathcal{O}^{\text{Classical}}} \text{ outputs } f(x)] \geq 2/3.$$

► **Definition 5.** The quantum query complexity  $Q(f)$  of  $f$  is the smallest integer  $q$  such that there exists a quantum algorithm  $\mathcal{A}^\mathcal{O}$  performing  $q$  queries to  $\mathcal{O}$  satisfying:

$$\forall x \in S, \Pr[\mathcal{A}_x^{\mathcal{O}} \text{ outputs } f(x)] \geq 2/3.$$

## 2.2 Hardness of distinguishing a random permutation from a random function with small range

Our proof will use a quantum lower bound on distinguishing a random permutation from a random function with small range proven in [11]. Following this paper, we define, for any  $r \in [n]$ , the following distribution  $D_r$  on functions from  $[n]$  to  $[n]$  from which can be sampled as follows.

- Draw a random function  $g$  from  $[n] \rightarrow [r]$ .
- Draw a random injective function  $h$  from  $[r] \rightarrow [n]$ .
- Output the composition  $h \circ g$ .

Notice that any function  $f$  drawn from  $D_r$  is of small range and satisfies  $|Im(f)| \leq r$ . Let also  $D_{\text{perm}}$  be the uniform distribution on permutations on  $[n]$ . Zhandry's lower bound can be stated as follows:

► **Proposition 6** ([11]). *There exists an absolute constant  $\Lambda$  such that for any  $r \in [n]$  and any quantum query algorithm  $\mathcal{B}^\circ$  performing at most  $\lceil \Lambda r^{1/3} \rceil$  queries to  $\mathcal{O}$ :*

$$\forall b \in \{0, 1\}, \left| \mathbb{E}_{\pi \leftarrow D_{\text{perm}}} \Pr[\mathcal{B}^{\circ\pi} \text{ outputs } b] - \mathbb{E}_{C \leftarrow D_r} \Pr[\mathcal{B}^{\circ C} \text{ outputs } b] \right| \leq \frac{2}{27}.$$

This is obtained immediately by combining Theorem 8 and Lemma 1 of [11]<sup>1</sup>.

### 3 Proving our main theorem

The goal of this section is to prove Theorem 3. Fix a function  $f : S \rightarrow \{0, 1\}$  where  $S \subseteq [M]^n$  with  $Q(f) = q$ . This means there exists a quantum query algorithm  $\mathcal{A}^\circ$  performing  $q$  queries to  $\mathcal{O}$  such that

$$\forall x \in S, \Pr[\mathcal{A}^{\circ x} \text{ outputs } f(x)] \geq 2/3.$$

We first amplify the success probability to  $20/27$ .

► **Lemma 7.** *There exists a quantum query algorithm  $\mathcal{A}_3^\circ$  that performs  $3q$  queries to  $\mathcal{O}$  such that*

$$\forall x \in S, \Pr[\mathcal{A}_3^{\circ x} \text{ outputs } f(x)] \geq \frac{20}{27}.$$

**Proof.**  $\mathcal{A}_3^\circ$  will consist of the following: run  $\mathcal{A}^\circ$  independently 3 times and take the output that occurs the most. For each  $x$ , each run of  $\mathcal{A}^{\circ x}$  outputs  $f(x)$  with probability greater than  $2/3$ . The probability that the correct  $f(x)$  appears at least twice out of the 3 results is therefore greater than  $\frac{8}{27} + 3 \cdot \frac{4}{27} = \frac{20}{27}$ . ◀

Using the fact that  $f$  is permutation symmetric, we get the following corollary:

► **Corollary 8.**

$$\forall x \in S, \forall \pi \in S_n, \Pr[\mathcal{A}_3^{\circ x \circ \pi} \text{ outputs } f(x)] = \Pr[\mathcal{A}_3^{\circ x \circ \pi} \text{ outputs } f(x \circ \pi)] \geq \frac{20}{27}.$$

<sup>1</sup> Equivalently, this is obtained immediately by combining Lemma 3.2 and Lemma 3.4 from the arXiv version [quant-ph:1312.1027](https://arxiv.org/abs/1312.1027).

### 3.1 Looking at a small number of indices of $x$

The main idea of the proof is to show that  $\mathcal{A}_3$  will output  $f(x)$  with high probability when replacing queries to  $\mathcal{O}_x$  with queries to  $\mathcal{O}_{x \circ C}$  for  $C$  chosen uniformly from  $D_r$  for some  $r = \Theta(Q^3(f))$ . First notice that for any  $x : [n] \rightarrow [M]$  and any  $g : [n] \rightarrow [n]$ , it is possible to apply  $\mathcal{O}_{x \circ g}$  with 2 calls to  $\mathcal{O}_g$  and 1 call to  $\mathcal{O}_x$  with the following procedure:

$$|i\rangle|j\rangle|0\rangle \rightarrow |i\rangle|j\rangle|g(i)\rangle \rightarrow |i\rangle|j + (x \circ g)(i)\rangle|g(i)\rangle \rightarrow |i\rangle|j + (x \circ g)(i)\rangle|0\rangle$$

where we respectively apply  $\mathcal{O}_g$  on registers (1, 3);  $\mathcal{O}_x$  on registers (3, 2) and  $\mathcal{O}_g^\dagger$  on registers (1, 3).

Therefore, for any fixed (and known)  $x$ , for any function  $g : [n] \rightarrow [n]$ , we can look at  $\mathcal{A}_3^{\mathcal{O}_{x \circ g}}$  as a quantum query algorithm that queries  $\mathcal{O}_g$ . In other words, for each  $x \in S$ , there is a quantum query algorithm  $\mathcal{B}_x^{\mathcal{O}_g}$  such that  $\mathcal{B}_x^{\mathcal{O}_g} = \mathcal{A}^{\mathcal{O}_{x \circ g}}$  for any function  $g : [n] \rightarrow [n]$ . Notice also that since a query to  $\mathcal{O}_{x \circ g}$  is done by doing 2 queries to  $\mathcal{O}_g$ , we have that  $\mathcal{B}^{\mathcal{O}}$  uses twice as many queries than  $\mathcal{A}_3^{\mathcal{O}}$ .

We can now prove our main proposition that shows that we can compute  $f(x)$  by looking only at  $x \circ C$  for a random  $C$  with  $|Im(C)| \leq r$ .

► **Proposition 9.** *Let  $f : [M]^n \rightarrow \{0, 1\}$  with  $Q(f) = q$  and  $r = \lceil 216q^3\Lambda^{-3} \rceil$  where  $\Lambda$  is the absolute constant from Proposition 6.*

$$\forall x \in S, \mathbb{E}_{C \leftarrow D_r} \Pr[\mathcal{A}_3^{\mathcal{O}_{x \circ C}} \text{ outputs } f(x)] \geq 2/3.$$

**Proof.** For each  $x \in S$ , we consider the algorithm  $\mathcal{B}_x^{\mathcal{O}}$  described above. Recall that for all  $g : [n] \rightarrow [n]$ ,  $\mathcal{B}_x^{\mathcal{O}_g} = \mathcal{A}_3^{\mathcal{O}_{x \circ g}}$ . Since  $\mathcal{A}_3^{\mathcal{O}}$  uses  $3q$  queries,  $\mathcal{B}_x^{\mathcal{O}}$  uses  $6q$  queries. We first consider the case where  $g$  is a random permutation. Using Corollary 8:

$$\forall x \in S, \mathbb{E}_{\pi \leftarrow D_{\text{perm}}} \Pr[\mathcal{B}_x^{\mathcal{O}_\pi} \text{ outputs } f(x)] = \mathbb{E}_{\pi \leftarrow D_{\text{perm}}} \Pr[\mathcal{A}_3^{\mathcal{O}_{x \circ \pi}} \text{ outputs } f(x)] \geq \frac{20}{27}$$

Using the lower bound of Proposition 6 noticing that  $6q \leq \Lambda r^{1/3}$ , we have

$$\forall x \in S, \left| \mathbb{E}_{\pi \leftarrow D_{\text{perm}}} \Pr[\mathcal{B}_x^{\mathcal{O}_\pi} \text{ outputs } f(x)] - \mathbb{E}_{C \leftarrow D_r} \Pr[\mathcal{B}_x^{\mathcal{O}_C} \text{ outputs } f(x)] \right| \leq \frac{2}{27}.$$

which gives us

$$\forall x \in S, \mathbb{E}_{C \leftarrow D_r} \Pr[\mathcal{B}_x^{\mathcal{O}_C} \text{ outputs } f(x)] \geq \frac{20}{27} - \frac{2}{27} = 2/3.$$

Since for each  $x \in S$ ,  $\mathcal{B}_x^{\mathcal{O}_C} = \mathcal{A}_3^{\mathcal{O}_{x \circ C}}$ , we can therefore conclude

$$\forall x \in S, \mathbb{E}_{C \leftarrow D_r} \Pr[\mathcal{A}_3^{\mathcal{O}_{x \circ C}} \text{ outputs } f(x)] \geq 2/3. \quad \blacktriangleleft$$

### 3.2 Constructing a classical query algorithm for $f$

We can now use the above proposition to prove our main theorem.

► **Theorem 2 (Restated).** *For any permutation symmetric function  $f$  of the first type,  $R(f) \leq O(Q^3(f))$ .*

**Proof.** Fix a function  $f : S \rightarrow \{0, 1\}$  where  $S \subseteq [M]^n$  with  $Q(f) = q$ . This means there exists a quantum query algorithm  $\mathcal{A}^{\mathcal{O}}$  performing  $q$  queries to  $\mathcal{O}$  such that

$$\forall x \in S, \Pr[\mathcal{A}^{\mathcal{O}_x} \text{ outputs } f(x)] \geq 2/3.$$

We construct a randomized algorithm that performs  $r = \lceil 216q^3\Lambda^{-3} \rceil$  classical queries to  $\mathcal{O}_x^{\text{Classical}}$  as follows:

1. Choose a random  $C$  according to distribution  $D_r$ .
2. Query  $\mathcal{O}_x^{\text{Classical}}$  to get all values  $x_i$  for  $i \in \text{Im}(C)$ . This requires  $|\text{Im}(C)| \leq r$  queries to  $\mathcal{O}_x^{\text{Classical}}$ . These queries fully characterize the function  $x \circ C$ , hence the quantum unitary  $\mathcal{O}_{x \circ C}$ .
3. From  $\mathcal{A}^\circ$ , construct the quantum algorithm  $\mathcal{A}_3^\circ$  as in Lemma 7. Recall that  $\mathcal{A}_3^\circ$  just consists of applying  $\mathcal{A}^\circ$  independently 3 times and output the majority outcome.
4. We consider  $\mathcal{A}_3^{\circ_{x \circ C}}$  as a quantum unitary circuit acting on  $t$  qubits. At each step of the algorithm, we store the  $2^t$  amplitudes. When  $\mathcal{O}_{x \circ C}$  is called, we use its representation from step 2 to calculate its action on the  $2^t$  amplitudes. Other parts of  $\mathcal{A}_3^{\circ_{x \circ C}}$  are treated the same way. While this uses a lot of computing power, it does not require any queries to  $\mathcal{O}_x^{\text{Classical}}$  or  $\mathcal{O}_x$  other than those used at step 2.

Step 4 outputs the same output distribution than the quantum algorithm  $\mathcal{A}_3^{\circ_{x \circ C}}$ . Using Proposition 9, for all  $x \in S$ , this algorithm outputs  $f(x)$  with probability at least  $2/3$ , which implies

$$R(f) \leq r = \lceil 216Q^3(f)\Lambda^{-3} \rceil. \quad \blacktriangleleft$$

Notice that after step 2, it is not possible to just compute  $f(x \circ C)$ , and try to show that it is equal to  $f(x)$  since we don't even always have  $x \circ C \in S$ . This is yet another example in query complexity where we use the behavior of a query algorithm on inputs not necessarily in the domain of  $f$ .

## 4 Conclusion

This result extends the class of functions for which we can show a polynomial relationship between the quantum and the randomized query complexity and improves the polynomial in general for permutation symmetric functions.

The first obvious open question is to close the gap between the best known speed-up for permutation symmetric function - which is quadratic - and the cubic lower bound obtained in this paper. Another open question is to see if such techniques can be extended to the case where the domain  $S$  is permutation symmetric, which implies the case of total functions. While the techniques seem specific to permutation symmetric functions, using a more powerful lower bound or considering inputs  $x$  in superposition (as in [12]) could give interesting results.

Also, we are currently extending those techniques to study the behavior of uniformly random inputs  $x$  in particular in the context of the quantum random oracle model. Here, we are interested in the power of quantum attacks on a cryptographic scheme while performing quantum queries to a uniformly random function. This technique seems promising to show that for many attacks, these quantum queries can be replaced with classical queries in the same way as in our steps 2-4.

---

## References

- 1 Scott Aaronson and Andris Ambainis. The Need for Structure in Quantum Speedups. *Theory of Computing*, 10(6):133–166, 2014. doi:10.4086/toc.2014.v010a006.
- 2 Scott Aaronson and Yaoyun Shi. Quantum Lower Bounds for the Collision and the Element Distinctness Problems. *J. ACM*, 51(4):595–605, July 2004. doi:10.1145/1008731.1008735.
- 3 A. Ambainis. Understanding Quantum Algorithms via Query Complexity. *ArXiv e-prints*, December 2017. arXiv:1712.06349.

- 4 Andris Ambainis. Polynomial Degree and Lower Bounds in Quantum Complexity: Collision and Element Distinctness with Small Range. *Theory of Computing*, 1(3):37–46, 2005. doi:10.4086/toc.2005.v001a003.
- 5 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.
- 6 Charles H. Bennett, Ethan Bernstein, Gilles Brassard, and Umesh Vazirani. Strengths and Weaknesses of Quantum Computing. *SIAM Journal of Computing*, 26(5):1510–1523, October 1997. doi:10.1137/S0097539796300933.
- 7 D. Deutsch and R. Jozsa. Rapid solution of problems by quantum computation. *Proceedings of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, 439(1907):553–558, 1992. doi:10.1098/rspa.1992.0167.
- 8 Samuel Kutin. Quantum Lower Bound for the Collision Problem with Small Range. *Theory of Computing*, 1(2):29–36, 2005. doi:10.4086/toc.2005.v001a002.
- 9 Peter W. Shor. Algorithms for Quantum Computation: Discrete Logarithms and Factoring. In *IEEE Symposium on Foundations of Computer Science*, pages 124–134, 1994. URL: [citeseer.ist.psu.edu/14533.html](http://citeseer.ist.psu.edu/14533.html).
- 10 Daniel R. Simon. On the Power of Quantum Cryptography. In *35th Annual Symposium on Foundations of Computer Science, Santa Fe, New Mexico, USA, 20-22 November 1994*, pages 116–123, 1994. doi:10.1109/SFCS.1994.365701.
- 11 Mark Zhandry. A Note on the Quantum Collision and Set Equality Problems. *Quantum Info. Comput.*, 15(7-8):557–567, May 2015. URL: <http://dl.acm.org/citation.cfm?id=2871411.2871413>.
- 12 Mark Zhandry. How to Record Quantum Queries, and Applications to Quantum Indifferentiability. *IACR Cryptology ePrint Archive*, 2018:276, 2018. URL: <https://eprint.iacr.org/2018/276>.





# Adaptive Boolean Monotonicity Testing in Total Influence Time

Deeparnab Chakrabarty<sup>1</sup>

Dartmouth College, Hanover, NH 03755, USA

deeparnab@dartmouth.edu

C. Seshadhri<sup>2</sup>

University of California, Santa Cruz, CA 95064, USA

sesh@ucsc.edu

---

## Abstract

Testing monotonicity of a Boolean function  $f : \{0, 1\}^n \rightarrow \{0, 1\}$  is an important problem in the field of property testing. It has led to connections with many interesting combinatorial questions on the directed hypercube: routing, random walks, and new isoperimetric theorems. Denoting the proximity parameter by  $\varepsilon$ , the best tester is the non-adaptive  $\tilde{O}(\varepsilon^{-2}\sqrt{n})$  tester of Khot-Minzer-Safra (FOCS 2015). A series of recent results by Belovs-Blais (STOC 2016) and Chen-Waingarten-Xie (STOC 2017) have led to  $\tilde{\Omega}(n^{1/3})$  lower bounds for *adaptive* testers. Reducing this gap is a significant question, that touches on the role of adaptivity in monotonicity testing of Boolean functions.

We approach this question from the perspective of *parametrized property testing*, a concept recently introduced by Pallavoor-Raskhodnikova-Varma (ACM TOCT 2017), where one seeks to understand performance of testers with respect to parameters other than just the size. Our result is an adaptive monotonicity tester with one-sided error whose query complexity is  $O(\varepsilon^{-2}\mathbf{I}(f)\log^5 n)$ , where  $\mathbf{I}(f)$  is the total influence of the function. Therefore, adaptivity provably helps monotonicity testing for low influence functions.

**2012 ACM Subject Classification** Theory of computation  $\rightarrow$  Streaming, sublinear and near linear time algorithms

**Keywords and phrases** Property Testing, Monotonicity Testing, Influence of Boolean Functions

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2019.20

## 1 Introduction

Monotonicity testing of Boolean functions  $f : \{0, 1\}^n \rightarrow \{0, 1\}$  is a classic problem in property testing that has been extensively studied over the last two decades [11, 8, 6, 2, 4, 3, 9, 1, 5]. Consider the coordinate-wise partial order over  $\{0, 1\}^n$ , denoted by  $\prec$ . A function  $f$  is monotone if  $\forall x, y \in \{0, 1\}^n$  where  $x \prec y$ ,  $f(x) \leq f(y)$ . The distance between two functions  $f, g$  is  $\Pr_x[f(x) \neq g(x)]$ , where  $x$  is u.a.r in  $\{0, 1\}^n$ . The distance of  $f$  to monotonicity is the minimum distance of  $f$  to a monotone  $g$ . Typically, we say that  $f$  is  $\varepsilon$ -far if the distance to monotonicity is at least  $\varepsilon$ .

The goal of monotonicity testing is to design efficient ( $\text{poly}(n)$ ) randomized procedures that distinguish monotone functions from those that are far from monotone. Formally, given query access to  $f$  and a proximity parameter  $\varepsilon > 0$ , a monotonicity tester is a (randomized) procedure that accepts if  $f$  is monotone, and rejects if  $f$  is  $\varepsilon$ -far from being monotone. Both

---

<sup>1</sup> Supported by NSF CCF-1813165.

<sup>2</sup> Supported by NSF TRIPODS CCF-1740850 and NSF CCF-1813165.



the above guarantees should hold with probability at least  $2/3$ . A tester has one-sided error if it always accepts a monotone function, or equivalently, always provides a certificate of rejection. A tester is called *non-adaptive* if all the queries can be made without seeing any of the answers. On the other hand, if the tester’s queries depend on previous answers, the tester is adaptive. As we explain below, the power of adaptivity is the central open question in Boolean monotonicity testing.

Raskhodnikova [11], Goldreich et al. [8], and Dodis et al. [6] initiated the study of monotonicity testing by describing a  $O(n/\varepsilon)$ -query non-adaptive, one-sided tester. Their “edge tester” repeats this simple procedure  $O(n/\varepsilon)$  times: sample a random edge of the hypercube and test for a monotonicity violation among its endpoints. Chakrabarty and Seshadhri [2] described the first  $o(n)$ -query tester, by proving connections between monotonicity testing and directed isoperimetry theorems. Their “path tester” performs random walks on the directed hypercube. Chen-Servedio-Tan [4] gave a tighter analysis to get an  $\tilde{O}(n^{5/6}\varepsilon^{-4})$  bound. In a remarkable result, Khot, Minzer, and Safra [9] (henceforth KMS) prove that the path tester works in  $\tilde{O}(\sqrt{n}/\varepsilon^2)$  queries. This was achieved by a deep isoperimetric result, a directed analog of Talagrand’s isoperimetry theorem [12].

Fischer et al. [7] had proven an  $\Omega(\sqrt{n})$  lower bound for non-adaptive, one-sided testers. Using sophisticated techniques, Chen-De-Servedio-Tan [3] proved an  $\Omega(n^{1/2-\delta})$  (for any fixed  $\delta > 0$ ) lower bound for non-adaptive, two-sided testers. All in all, this nearly resolves the non-adaptive complexity of monotonicity testing.

In a major recent advance, Belovs-Blais [1] proved the first polynomial lower bound of  $\tilde{\Omega}(n^{1/4})$  for any *adaptive* tester. This was further improved by Chen-Waingarten-Xie [5] to  $\tilde{\Omega}(n^{1/3})$ . These are highly non-trivial results. Belovs-Blais also gave a  $O(\log n)$  query adaptive monotonicity tester for the class of regular linear threshold functions (LTFs), the hard functions of Chen et al. [3]. This underscores the challenge of proving adaptive lower bounds and leaves a tantalizing open question. *Can adaptivity always help in Boolean monotonicity testing over the hypercube?*

We approach this question from the perspective of *parametrized property testing*, a concept recently introduced by Pallavoor-Raskhodnikova-Varma [10]. One seeks to understand the performance of testers with respect to parameters other than just the size of the domain. If one can beat existing lower bounds (for some settings of the parameters), this gives insight into the hardest functions for monotonicity testing.

Our main result is the existence of adaptive monotonicity testers parametrized by the total influence,  $\mathbf{I}(f)$ . Letting  $\mathcal{D}$  be the uniform distribution over all pairs  $(x, y)$  at Hamming distance 1,  $\mathbf{I}(f) := n \cdot \Pr_{(x,y) \sim \mathcal{D}}[f(x) \neq f(y)]$ .

► **Theorem 1.** *Consider the class of functions  $f : \{0, 1\}^n \rightarrow \{0, 1\}$  with total influence at most  $I$ . There exists a one-sided, adaptive monotonicity tester for this class with query complexity  $O(I\varepsilon^{-2} \log^5 n)$ .*

A claim of KMS (Theorem 9.1 in [9]) implies that if  $\mathbf{I}(f) > 6\sqrt{n}$ , then the edge tester is itself a  $O(n/\mathbf{I}(f))$  tester. Combined with Theorem 1, one gets an adaptive  $\tilde{O}(\min(\mathbf{I}(f), n/\mathbf{I}(f))\varepsilon^{-2})$ -query tester. The trade-off point is basically  $\sqrt{n}$ , the maximum possible influence of a monotone function.

## 1.1 Perspective on Theorem 1

- When  $\mathbf{I}(f) \ll \sqrt{n}$ , Theorem 1 beats the lower bounds of [7, 3]. Indeed, the lower bound of Fischer et al. [7] holds for *constant* influence functions, and so adaptivity is crucial for a result like Theorem 1. As mentioned earlier, it was known that adaptivity helps for the structured class of regular LTFs [1]. What is intriguing about Theorem 1 is that no strong structural assumption is required to get  $o(\sqrt{n})$  query testers.

- All adaptive lower bound constructions have functions with influence  $\Theta(\sqrt{n})$  [1, 5]. In light of Theorem 1, this is perhaps no accident. Theorem 1 shows that this is the hard regime for monotonicity testing.
- The  $o(n)$  non-adaptive testers are obtained by directed random walks in the hypercube. One starts at a random  $x$  and walks “up” (consistent with the partial order) the hypercube to reach some  $y$ . Then, the tester compares  $f(x), f(y)$  to detect non-monotonicity. (The intermediate vertices in the random walk are not queried, and this is crucial for getting the  $o(n)$  bound.) The algorithm of Theorem 1 is fundamentally different; it performs standard, *undirected* random walks on the hypercube. The endpoints might not even be comparable, so only querying these is of no use. This is exactly where adaptivity helps, since we can perform binary search to find violations in this path. This leads to a better monotonicity tester for functions with influence  $o(\sqrt{n})$ .

## 1.2 Proof Idea

The  $o(n)$  testers of [2, 4, 9] perform random walks on the directed hypercube with orientation corresponding to the partial order, and query the function at the endpoints of this walk. Their analysis shows that if the function is  $\varepsilon$ -far from being monotone, then there is a significant probability of encountering a violation. At a high level, one of the main observations of [2, 4, 9] is that “longer” walks lead to higher chances of catching a violation. However, the vast majority of vertices in the hypercube have Hamming weight  $n/2 \pm \Theta(\sqrt{n})$ . For the purposes of monotonicity testing, one can assume that vertices outside these middle layers do not participate in any violations to monotonicity. Each step of a directed walk increments the Hamming weight, and thus the walk length can be at most  $\Theta(\sqrt{n})$ . This  $\sqrt{n}$  is precisely what shows up in the final bound of KMS.

Our insight is that one can perform an analogous analysis as above for random walks on the undirected hypercube  $H_n$ . These walks can have length  $\omega(\sqrt{n})$ . Suppose we perform an  $\ell$ -step random walk (on  $H_n$ ) from  $x$  that ends at  $y$ . Note that with high probability,  $x$  and  $y$  will not be comparable. Nonetheless, if  $f(x) \neq f(y)$ , then the walk has passed through an influential edge. The power of adaptivity is that we can find such an influential edge through binary search. This idea of using binary search is indeed also present in a  $O(\log n)$ -query algorithm of Belovs and Blais [1] for adaptive monotonicity testers for regular linear threshold functions.

However, we are not interested in finding an influential edge but rather a *violated* edge. Fix a violated edge  $(u, v)$ . Our insight is to lower bound the probability that  $(u, v)$  is the *unique* influential edge in the undirected random walk. In this scenario, binary search is guaranteed to find a violation. There are two opposing forces here. First, the probability that  $(u, v)$  at all appears in the random walk grows as  $\ell/n$ . On the other hand, the probability that this is the unique influential edge decreases with  $\ell$ ; indeed, this probability depends on the influence of  $f$ . A fairly simple calculation shows that for all but an  $\ell \mathbf{I}(f)/n$  fraction of “bad” vertices, an  $\ell$ -length from a vertex encounters no influential edge with constant probability. Putting these together, we see that setting  $\ell \sim n/\mathbf{I}(f)$  would lead to a good probability of catching a violation; note that if  $\mathbf{I}(f) \ll \sqrt{n}$ , the desired length of the walk would indeed be  $\gg \sqrt{n}$ .

The only fly in the above ointment is if all the violated edges were concentrated on the bad vertices. This is where we invoke the connection between distance to monotonicity and isoperimetry; the directed Talagrand inequality of KMS essentially precludes this scenario by showing that violated edges need to be “spread apart”. The actual math is more subtle. KMS gives a trade-off between the concentration of the violated edges and the number of

violated edges. If all violated edges are incident only on a few vertices, then there must be a lot of violated edges. The latter is where the edge-tester, which is nothing but a length 1 random walk, works. This analysis is similar to what is done in KMS.

### 1.3 Preliminaries

We use  $H_n$  to denote the standard (undirected) hypercube graph on  $\{0, 1\}^n$ , where all pairs at Hamming distance 1 are connected. An edge  $(x, y)$  of  $H_n$  is influential, if  $f(x) \neq f(y)$ . The number of influential edges is precisely  $\mathbf{I}(f) \cdot 2^{n-1}$ . An influential edge  $(x, y)$  is a violating edge if  $x \prec y$ ,  $f(x) = 1$ , and  $f(y) = 0$ . Our tester will perform random walks of  $H_n$ . Note that  $H_n$  is regular, so this is a symmetric Markov Chain.

We crucially use the central result of KMS, which is obtained as a consequence of the directed analogue of Talagrand's isoperimetric theorem.

► **Lemma 2** (Lemma 7.1 in [9], paraphrased). *Given any Boolean function  $f : \{0, 1\}^n \rightarrow \{0, 1\}$  that is  $\varepsilon$ -far from being monotone, there exists a subgraph  $G = (A, B, E)$  of the hypercube and parameters  $\sigma \in (0, 1)$ ,  $d \in \mathbb{N}$  such that*

- *Each edge  $(a, b) \in E$  with  $a \in A$  and  $b \in B$  is a violating edge.*
- *The degree of each vertex in  $B$  is exactly  $d$  and the degree of each vertex in  $A$  is at most  $2d$ .*
- *$|B| = \sigma \cdot 2^n$ .*
- *$\sigma^2 d = \Omega(\varepsilon^2 / \log^4 n)$ .*

## 2 Tester and Analysis

Let  $f : \{0, 1\}^n \rightarrow \{0, 1\}$  be a Boolean function over the hypercube with total influence  $\mathbf{I}(f)$ .

---

**Algorithm 1** Adaptive Monotonicity Tester for Boolean Functions.

---

**Input:** A Boolean function  $f : \{0, 1\}^n \rightarrow \{0, 1\}$  and a parameter  $\varepsilon \in (0, 1)$

1. Choose  $k \in_R \{0, 1, 2, \dots, \lceil \log n \rceil\}$  uniformly at random. Set  $\ell := 2^k$ .
  2. Choose  $x \in \{0, 1\}^n$  uniformly at random.
  3. Perform an  $\ell$ -length random walk  $p$  on  $H_n$  starting from  $x$  to reach  $y \in \{0, 1\}^n$ .
  4. If  $f(x) \neq f(y)$ :
    - a. Perform binary search on  $p$  to find an influential edge  $(u, v) \in p$ .
    - b. REJECT if  $(u, v)$  is a monotonicity violation.
  5. ACCEPT.
- 

Theorem 1 follows directly from the following theorem by repeating the above subroutine the appropriate number of times. Note that the subroutine above does not need to know  $\mathbf{I}(f)$ .

► **Theorem 3.** *If  $f$  is  $\varepsilon$ -far from being monotone, then the algorithm described in Algorithm 1 rejects with probability  $\Omega\left(\frac{\varepsilon^2}{\mathbf{I}(f) \log^5 n}\right)$ .*

► **Definition 4.** Given a positive integer  $\ell$ , a vertex  $x \in \{0, 1\}^n$  is denoted  $\ell$ -sticky if an  $\ell$ -length random walk from  $x$  on  $H_n$  contains no influential edges with probability  $\geq 1/2$ . A vertex is called non- $\ell$ -sticky otherwise. An edge is  $\ell$ -sticky if both endpoints are  $\ell$ -sticky.

► **Observation 5.** *If a vertex  $x$  is  $\ell$ -sticky and  $\ell' < \ell$ , then  $x$  is  $\ell'$ -sticky as well.*

► **Lemma 6.** *The fraction of non- $\ell$ -sticky vertices of a hypercube is at most  $\frac{2\ell \cdot \mathbf{I}(f)}{n}$ .*

**Proof.** Given  $x \in \{0, 1\}^n$  and a positive integer  $\ell > 0$ , define the random variable  $Z_{x,\ell}$  that is the number of influential edges in a random walk of length  $\ell$  starting from  $x$ . Therefore,  $x$  is non- $\ell$ -sticky iff  $\Pr[Z_{x,\ell} > 0] > 1/2$ . Let  $N$  denote the set of non- $\ell$ -sticky vertices.

Since  $Z_{x,\ell}$  is non-negative and integer valued we get  $\Pr[Z_{x,\ell} > 0] \leq \mathbf{E}[Z_{x,\ell}]$ .

$$|N|/2^n < \frac{2}{2^n} \sum_{x \in N} \Pr[Z_{x,\ell} > 0] < \frac{2}{2^n} \sum_{x \in \{0,1\}^n} \Pr[Z_{x,\ell} > 0] \leq \frac{2}{2^n} \sum_{x \in \{0,1\}^n} \mathbf{E}[Z_{x,\ell}] \quad (1)$$

The RHS above is precisely twice the expected number of influential edges encountered in an  $\ell$ -length random walk starting from the uniform distribution on  $H_n$ . Let  $\mathcal{P}_\ell$  denote the uniform distribution on  $\ell$ -length paths in  $H_n$ . For  $p \sim \mathcal{P}_\ell$ ,  $p_t$  denotes the  $t$ th edge in  $p$ , and let  $\chi(e)$  be the indicator for edge  $e$  being influential. The RHS of (1) is equal to  $2\mathbf{E}_{p \sim \mathcal{P}_\ell}[\sum_{t \leq \ell} \chi(p_t)] = 2 \sum_{t \leq \ell} \mathbf{E}_{p \sim \mathcal{P}_\ell}[\chi(p_t)]$ . Since the uniform distribution is stationary for random walks on  $H_n$ , the distribution induced on  $p_t$  is the uniform distribution on edges in  $H_n$ . Thus,  $\mathbf{E}_{p \sim \mathcal{P}_\ell}[\chi(p_t)] = \mathbf{I}(f)/n$  and the RHS of (1) is  $2\ell \cdot \mathbf{I}(f)/n$ . ◀

For any integer  $\ell > 0$ , let  $F_\ell$  be the set of  $\ell$ -sticky violating edges. That is,

$$F_\ell := \{(x, y) \in H_n : (x, y) \text{ is violating and, } x, y \text{ are } \ell\text{-sticky}\}$$

► **Lemma 7.** *If  $\ell$  is the length of the random walk chosen in Step 1, then Algorithm 1 rejects with probability  $\Omega\left(\frac{\ell}{n} \cdot \frac{|F_\ell|}{2^n}\right)$ .*

**Proof.** Fix an edge  $(u, v) \in F_\ell$ . Let  $p = (e_1, \dots, e_\ell)$  be the edges of a random walk  $p$ . Each edge  $e_t$  for  $1 \leq t \leq \ell$  is a uniformly sampled random edge of  $H_n$ . Therefore, for any fixed edge  $(u, v)$ ,  $\Pr[e_t = (u, v)] = \frac{1}{n \cdot 2^{n-1}}$ .

Now, given  $1 \leq t \leq \ell$ , define  $\mathcal{E}_{u,v}^t$  to be the event that the  $t$ th edge of  $p$  is  $(u, v)$  and no other edge of  $p$  is influential. Define  $\mathcal{E}_{u,v}$  to be the disjoint union  $\bigvee_{t=1}^\ell \mathcal{E}_{u,v}^t$ . Observe that for two distinct edges  $(u, v)$  and  $(u', v')$  in  $F_\ell$ , the events  $\mathcal{E}_{u,v}$  and  $\mathcal{E}_{u',v'}$  are disjoint. Therefore,

$$\Pr[\text{Algorithm 1 rejects for length } \ell] \geq \Pr\left[\bigvee_{(u,v) \in F_\ell} \mathcal{E}_{u,v}\right] = \sum_{(u,v) \in F_\ell} \Pr[\mathcal{E}_{u,v}] \quad (2)$$

The inequality follows since if  $\mathcal{E}_{u,v}$  occurs then the end points of  $p$  must have differing values and binary search on  $p$  will return the violation  $(u, v)$ . The equality follows since the events are mutually exclusive.

Consider the event  $\mathcal{E}_{u,v}^t$ . For this event to occur,  $e_t$  must be  $(u, v)$ . Consider the conditional probability  $\Pr[\mathcal{E}_{u,v}^t \mid e_t = (u, v)]$ . Let  $\mathcal{F}_u$  be the event that a  $(t-1)$ -length random walk from  $u$  contains no influential edges, and let  $\mathcal{F}_v$  be the event that an *independent*  $(\ell-t)$ -length random walk from  $v$  contains no influential edges.

► **Claim 8.**  $\Pr[\mathcal{E}_{u,v}^t \mid e_t = (u, v)] = \Pr[\mathcal{F}_u \wedge \mathcal{F}_v] = \Pr[\mathcal{F}_u] \cdot \Pr[\mathcal{F}_v]$

**Proof.** Conditioned on  $e_t = (u, v)$ , the distribution of the first  $(t-1)$  steps of the random walk is the uniform distribution of  $(t-1)$ -length paths that end at  $u$ . This is the same distribution of the  $(t-1)$ -length random walk starting from  $u$ . The distribution of the last  $(\ell-t)$  steps, conditioned on  $e_t = (u, v)$ , is the  $(\ell-t)$ -length random walk starting from  $v$ . ◀

Since  $(u, v)$  is an  $\ell$ -sticky edge, by Obs 5 and Definition 4,  $\Pr[\mathcal{F}_u] \geq 1/2$  and  $\Pr[\mathcal{F}_v] \geq 1/2$ . The proof is completed by plugging the following bound into (2).

$$\begin{aligned} \Pr[\mathcal{E}_{u,v}] &= \sum_{t=1}^{\ell} \Pr[\mathcal{E}_{u,v}^t] = \sum_{t=1}^{\ell} \Pr[e_t = (u, v)] \cdot \Pr[\mathcal{E}_{u,v}^t \mid e_t = (u, v)] \\ &= \frac{2}{n \cdot 2^n} \sum_{t=1}^{\ell} \Pr[\mathcal{E}_{u,v}^t \mid e_t = (u, v)] \geq \frac{\ell}{4n \cdot 2^n} \quad \blacktriangleleft \end{aligned}$$

We complete the proof of Theorem 3.

**Proof of Theorem 3.** Let  $G = (A, B, E)$ ,  $\sigma, d$  be as in Lemma 2.

**Case 1:**  $\frac{n\sigma}{16\mathbf{I}(f)} \leq 1$ .

In this case, we argue that the edge tester succeeds. More precisely, consider the setting of the algorithm described in Algorithm 1 that sets  $\ell = 1$ , that is, the tester checks whether a random edge of  $H_n$  is a violation. This setting occurs with probability  $\Omega(1/\log n)$ . The number of violated edges is at least  $\sigma d 2^n$ , the number of edges in  $G = (A, B, E)$  (as defined in Lemma 2). Since  $\sigma^2 d = \Omega(\varepsilon^2 / \log^4 n)$ ,  $|E| = \Omega(\frac{\varepsilon^2}{\log^4 n} \cdot \frac{2^n}{\sigma})$ . Since the number of edges of the hypercube is  $\Theta(n 2^n)$ , we get that the probability Algorithm 1 obtains a violation is  $\Omega(\frac{\varepsilon^2}{\log^5 n} \cdot \frac{1}{n\sigma})$ . By assumption,  $\frac{n\sigma}{16\mathbf{I}(f)} \leq 1$ , so this probability is  $\Omega(\frac{\varepsilon^2}{\mathbf{I}(f) \log^5 n})$ .

**Case 2:**  $\frac{n\sigma}{16\mathbf{I}(f)} > 1$ .

In this case, there exists a non-negative power of 2 (call it  $\ell^*$ ) such that  $\frac{\sigma}{16} < \frac{\ell^* \mathbf{I}(f)}{n} \leq \frac{\sigma}{8}$ . Let  $A'$  and  $B'$  be the subset of  $A$  and  $B$  that are  $\ell^*$ -sticky. Let  $E' \subseteq E$  be the edges with end points in  $A'$  and  $B'$ . Note that any edge of  $E' \subseteq F_{\ell^*}$ . By Lemma 6, we get that the fraction of non- $\ell^*$ -sticky vertices is at most  $2^{\ell^*} \cdot \mathbf{I}(f)/n \leq \sigma/4$ . Since the degree of any vertex in  $G$  in Lemma 2 is  $\leq 2d$ ,

$$|F_{\ell^*}| \geq |E'| \geq |E| - (2d) \cdot \frac{\sigma \cdot 2^n}{4} = \frac{\sigma d 2^n}{2}.$$

The probability the algorithm chooses  $\ell = \ell^*$  is  $1/\log n$ . Lemma 7 gives us

$$\begin{aligned} \Pr[\text{Algorithm rejects}] &\geq \frac{1}{\log n} \cdot \Pr[\text{Algorithm rejects} \mid \ell = \ell^*] \\ &\geq \frac{1}{\log n} \cdot \left( \frac{\ell^*}{n} \cdot \frac{|F_{\ell^*}|}{2^n} \right) \quad (\text{by Lemma 7}) \\ &\geq \frac{1}{\log n} \cdot \frac{\ell^*}{n} \cdot \frac{\sigma d}{2} \\ &\geq \frac{1}{\mathbf{I}(f)} \cdot \left( \frac{\sigma^2 d}{32 \log n} \right) \quad (\text{plugging } \ell^* \geq \sigma n / (16 \cdot \mathbf{I}(f))) \\ &= \Omega\left( \frac{\varepsilon^2}{\mathbf{I}(f) \log^5 n} \right) \quad (\text{by Lemma 2}) \quad \blacktriangleleft \end{aligned}$$

---

## References

- 1 Aleksandrs Belovs and Eric Blais. A Polynomial Lower Bound for Testing Monotonicity. In *Proceedings, ACM Symposium on Theory of Computing (STOC)*, 2016.
- 2 Deeparnab Chakrabarty and C. Seshadhri. An  $o(n)$  Monotonicity Tester for Boolean Functions over the Hypercube. *SIAM Journal on Computing (SICOMP)*, 45(2):461–472, 2014.

- 3 Xi Chen, Anindya De, Rocco A. Servedio, and Li-Yang Tan. Boolean Function Monotonicity Testing Requires (Almost)  $O(n^{1/2})$  Non-adaptive Queries. In *Proceedings, ACM Symposium on Theory of Computing (STOC)*, 2015.
- 4 Xi Chen, Rocco A. Servedio, and Li-Yang. Tan. New Algorithms and Lower Bounds for Monotonicity Testing. In *Proceedings, IEEE Symposium on Foundations of Computer Science (FOCS)*, 2014.
- 5 Xi Chen, Erik Waingarten, and Jinyu Xie. Beyond Talagrand: New Lower Bounds for Testing Monotonicity and Unateness. In *Proceedings, ACM Symposium on Theory of Computing (STOC)*, 2017.
- 6 Yevgeny Dodis, Oded Goldreich, Eric Lehman, Sofya Raskhodnikova, Dana Ron, and Alex Samorodnitsky. Improved Testing Algorithms for Monotonicity. *Proceedings, International Workshop on Randomization and Computation (RANDOM)*, 1999.
- 7 Eldar Fischer, Eric Lehman, Ilan Newman, Sofya Raskhodnikova, and Ronitt Rubinfeld. Monotonicity Testing over General Poset Domains. *Proceedings, ACM Symposium on Theory of Computing (STOC)*, 2002.
- 8 Oded Goldreich, Shafi Goldwasser, Eric Lehman, Dana Ron, and Alex Samordinsky. Testing Monotonicity. *Combinatorica*, 20:301–337, 2000.
- 9 Subhash Khot, Dor Minzer, and Muli Safra. On Monotonicity Testing and Boolean Isoperimetric Type Theorems. In *Proceedings, IEEE Symposium on Foundations of Computer Science (FOCS)*, 2015.
- 10 Ramesh Krishnan S. Pallavoor, Sofya Raskhodnikova, and Nithin Varma. Parameterized Property Testing of Functions. *ACM Transactions on Computation Theory (TOCT)*, 9(4), 2017.
- 11 Sofya Raskhodnikova. Monotonicity Testing. *Masters Thesis, MIT*, 1999.
- 12 Michel Talagrand. Isoperimetry, Logarithmic Sobolev inequalities on the Discrete Cube, and Margulis’ Graph Connectivity Theorem. *Geom. Func. Anal.*, 3(3):295–314, 1993.





# On Locality-Sensitive Orderings and Their Applications

**Timothy M. Chan**

Department of Computer Science, University of Illinois at Urbana-Champaign, USA  
tmc@illinois.edu

**Sariel Har-Peled<sup>1</sup>**

Department of Computer Science, University of Illinois at Urbana-Champaign, USA  
sariel@illinois.edu

**Mitchell Jones**

Department of Computer Science, University of Illinois at Urbana-Champaign, USA  
mfjones2@illinois.edu

---

## Abstract

For any constant  $d$  and parameter  $\varepsilon > 0$ , we show the existence of (roughly)  $1/\varepsilon^d$  orderings on the unit cube  $[0, 1]^d$ , such that any two points  $p, q \in [0, 1]^d$  that are close together under the Euclidean metric are “close together” in one of these linear orderings in the following sense: the only points that could lie between  $p$  and  $q$  in the ordering are points with Euclidean distance at most  $\varepsilon\|p - q\|$  from  $p$  or  $q$ . These orderings are extensions of the  $\mathcal{Z}$ -order, and they can be efficiently computed.

Functionally, the orderings can be thought of as a replacement to quadtrees and related structures (like well-separated pair decompositions). We use such orderings to obtain surprisingly simple algorithms for a number of basic problems in low-dimensional computational geometry, including (i) dynamic approximate bichromatic closest pair, (ii) dynamic spanners, (iii) dynamic approximate minimum spanning trees, (iv) static and dynamic fault-tolerant spanners, and (v) approximate nearest neighbor search.

**2012 ACM Subject Classification** Theory of computation  $\rightarrow$  Computational geometry

**Keywords and phrases** Approximation algorithms, Data structures, Computational geometry

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2019.21

**Acknowledgements** The authors thank the anonymous reviewers for their helpful suggestions.

## 1 Introduction

In this paper, we describe a technique that leads to new simple algorithms for a number of fundamental proximity problems in low-dimensional Euclidean spaces.

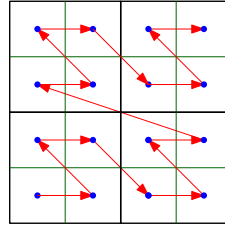
### Notation

The  $O$  notation hides constants that depends (usually exponentially) on  $d$ . Throughout, we assume (without loss of generality) that  $\varepsilon$  is a power of 2; that is  $\varepsilon = 2^{-E}$  for some positive integer  $E$ .

---

<sup>1</sup> Supported in part by NSF AF award CCF-1421231.



**Z-order**

Consider a point set  $P \subseteq [0, 1)^2$ , its quadtree, and a DFS traversal of this quadtree. One can order the points of  $P$  according to this traversal, resulting in an ordering  $\prec$  of the underlying set  $[0, 1)^2$ . The relation  $\prec$  is the ordering along a space filling mapping, known as the **Z-order**. Specifically, there is a bijection  $z$  from the unit interval  $[0, 1)$  to the unit square  $[0, 1)^2$  such that the ordering along the resulting “curve” is the Z-order. The Z-order mapping  $z$  is not continuous. Nevertheless, the Z-order mapping has the advantage of being easy to define. Indeed, given a real number  $\alpha \in [0, 1)$ , with the binary expansion  $\alpha = 0.x_1x_2x_3\dots$  (i.e.,  $\alpha = \sum_{i=1}^{\infty} x_i2^{-i}$ ), the Z-order mapping of  $\alpha$  is the point  $z(\alpha) = (0.x_2x_4x_6\dots, 0.x_1x_3x_5\dots)$ . Computing the Z-order or its inverse is quite easy, if one is allowed bitwise-logical operations – in particular, the ability to compute compressed quadtrees efficiently is possible only if such operations are available [19]. The approach extends to higher constant dimensions. The idea of using the Z-order can be traced back to the work of Gargantini [16], and it is widely used in databases and seems to improve performance in practice [24]. Once comparison by Z-order is available, building a compressed quadtree is no more than storing the points according to the Z-order, and this yields simple data structures for various problems. For example, Laio *et al.* [27] and Chan [7, 8, 9] applied the Z-order to obtain simple efficient algorithms for approximate nearest neighbor search and related problems.

**Shifting**

The Z-order (and quadtrees) does not preserve distance. That is, two points that are far away might be mapped to two close-together points, and vice versa. This problem is even apparent when using a grid, where points that are close together get separated into different grid cells. One way to get around this problem is to shift the grid (deterministically or randomly) [21]. The same approach works for quadtrees – one can shift the quadtree constructed for a point set several times such that for any pair of points in the quadtree, there will be a shift where the two points are in a cell of diameter that is  $O(1)$  times their distance. Improving an earlier work by Bern [3], Chan [6] showed that  $2\lceil d/2 \rceil + 1$  deterministic shifts are enough in  $d$  dimensions (a proof is reproduced in Appendix A.2). A somewhat similar shifting scheme was also suggested by Feige and Krauthgamer [14]. Random shifting of quadtrees underlines, for example, the approximation algorithm by Arora for Euclidean TSP [2].

By combining Z-order with shifting, both Chan [7] and Laio *et al.* [27] observed an extremely simple data structure for  $O(1)$ -approximate nearest neighbor search in constant dimensions: just store the points in Z-order for each of the  $2\lceil d/2 \rceil + 1$  shifts; given a query point  $q$ , find the successor and predecessor of  $q$  in the Z-order by binary search for each of the shifts, and return the closest point found. The data structure can be easily made dynamic to support insertions and deletions of points, and can also be adapted to find  $O(1)$ -approximate bichromatic closest pairs.

For approximate nearest neighbor (ANN) search, the  $O(1)$  approximation factor can be reduced to  $1 + \varepsilon$  for any fixed  $\varepsilon > 0$ , though the query algorithm becomes more involved [7] and unfortunately cannot be adapted to compute  $(1 + \varepsilon)$ -approximate bichromatic closest pairs. (In the monochromatic case, however, the approach can be adapted to find *exact* closest pairs, by considering  $O(1)$  successors and predecessors of each point [7].)

For other proximity-related problems such as spanners and approximate minimum spanning trees (MST), this approach does not seem to work as well: for example, the static algorithms in [9], which use the  $\mathcal{Z}$ -order, still require explicit constructions of compressed quadtrees and are not easily dynamizable.

### Main new technique: Locality Sensitive Orderings

For any given  $\varepsilon > 0$ , we show that there is a family of  $O((1/\varepsilon^d) \log(1/\varepsilon))$  orderings of  $[0, 1]^d$  with the following property: For any  $p, q \in [0, 1]^d$ , there is an ordering in the family such that all points lying between  $p$  and  $q$  in this ordering are within distance at most  $\varepsilon \|p - q\|$  from  $p$  or  $q$ . The order between two points can be determined efficiently using some bitwise-logical operations. See Theorem 10. We refer to these as **locality-sensitive orderings**. They generalize the previous construction of  $2\lceil d/2 \rceil + 1$  shifted copies of the  $\mathcal{Z}$ -order, which guarantees the stated property only for a large specific constant  $\varepsilon$  (dependent on  $d$ ). The property ensures, for example, that a  $(1 + \varepsilon)$ -approximate nearest neighbor of a point  $q$  can be found among the immediate predecessors and successors of  $q$  in these orderings.

### Applications

Locality-sensitive orderings immediately lead to simple algorithms for a number of problems, as listed below. Many of these results are significant simplification of previous work; some of the results are new.

- (a) **Approximate bichromatic closest pair.** Theorem 12 presents a data structure that maintains a  $(1 + \varepsilon)$ -approximate closest bichromatic pair for two sets of points in  $\mathbb{R}^d$ , with an update time of  $O(\log n)$ , for any fixed  $\varepsilon > 0$ , ignoring factors that depend on  $\varepsilon$  (roughly  $1/\varepsilon^d$ ). Previously, a general technique of Eppstein [12] can be applied in conjunction with a dynamic data structure for ANN, but the amortized update time increases by two  $\log n$  factors.
- (b) **Dynamic spanners.** For a parameter  $t \geq 1$  and a set of points  $P$  in  $\mathbb{R}^d$ , a graph  $G = (P, E)$  is a  $t$ -spanner for  $P$  if for all  $p, q \in P$ , there is a  $p$ - $q$  path in  $G$  of length at most  $t \|p - q\|$ . Static algorithms for spanners have been extensively studied in computational geometry. The dynamic problem appears tougher, and has also received much attention (see Table 1.1). We obtain a very simple data structure for maintaining a dynamic  $(1 + \varepsilon)$ -spanners in Euclidean space with an update (insertion and deletion) time of  $O(\log n)$  and having  $O(n)$  edges in total, for any fixed  $\varepsilon > 0$ , ignoring factors that depend on  $\varepsilon$  (roughly  $1/\varepsilon^d$ ). See Theorem 14. Although Gottlieb and Roditty [17] have previously obtained the same update time  $(1/\varepsilon^{O(d)}) \log n$ , their method requires much more intricate details. (Note that Gottlieb and Roditty's method more generally applies to spaces with bounded doubling dimension, but no simpler methods have been reported in the Euclidean setting.)

## 21:4 On Locality-Sensitive Orderings and Their Applications

■ **Table 1.1** Previous work on dynamic  $(1 + \varepsilon)$ -spanners in  $\mathbb{R}^d$ . Dependencies on  $\varepsilon$  (of the form  $1/\varepsilon^{O(d)}$ ) are omitted in the  $O$  bounds.

reference	insertion time	deletion time
Roditty [30]	$O(\log n)$	$O(n^{1/3} \log^{O(1)} n)$
Gottlieb and Roditty [18]	$O(\log^2 n)$	$O(\log^3 n)$
Gottlieb and Roditty [17]	$O(\log n)$	$O(\log n)$

■ **Table 1.2** Previous work on static  $k$ -vertex-fault-tolerant  $(1 + \varepsilon)$ -spanners in  $\mathbb{R}^d$ . Dependencies on  $\varepsilon$  (of the form  $1/\varepsilon^{O(d)}$ ) are omitted in the  $O$  bounds.

reference	# edges	degree	running time
Levcopoulos <i>et al.</i> [26]	$2^{O(k)} n$	$2^{O(k)}$	$O(n \log n + 2^{O(k)} n)$
	$O(k^2 n)$	unbounded	$O(n \log n + k^2 n)$
	$O(kn \log n)$	unbounded	$O(kn \log n)$
Lukovszki [28, 29]	$O(kn)$	$O(k^2)$	$O(n \log^{d-1} n + kn \log \log n)$
Czumaj and Zhao [11]	$O(kn)$	$O(k)$	$O(kn \log^d n + k^2 n \log k)$
H. Chan <i>et al.</i> [5]	$O(k^2 n)$	$O(k^2)$	$O(n \log n + k^2 n)$
Kapoor and Li [25]/Solomon [31]	$O(kn)$	$O(k)$	$O(n \log n + kn)$

- (c) **Dynamic approximate minimum spanning trees.** As is well-known [4, 19], a  $(1 + \varepsilon)$ -approximate Euclidean MST of a point set  $P$  can be computed from the MST of a  $(1 + \varepsilon)$ -spanner of  $P$ . In our dynamic spanner (and also Gottlieb and Roditty’s method [17]), each insertion/deletion of a point causes  $O(1)$  edge updates to the graph. Immediately, we thus obtain a dynamic data structure for maintaining a  $(1 + \varepsilon)$ -approximate Euclidean MST, with update time (ignoring dependencies on  $\varepsilon$ ) equal to that for the dynamic graph MST problem, which is currently  $O(\log^4 n / \log \log n)$  with amortization [22].
- (d) **Static and dynamic vertex-fault-tolerant spanners.** For parameters  $k, t \geq 1$  and a set of points  $P$  in  $\mathbb{R}^d$ , a  $k$ -vertex-fault-tolerant  $t$ -spanner is a graph  $G$  which is a  $t$ -spanner and for any  $P' \subseteq P$  of size at most  $k$ , the graph  $G \setminus P'$  remains a  $t$ -spanner for  $P \setminus P'$ . Fault-tolerant spanners have been extensively studied (see Table 1.2). Locality-sensitive orderings lead to a very simple construction for  $k$ -vertex-fault-tolerant  $(1 + \varepsilon)$ -spanners, with  $O(kn)$  edges, maximum degree  $O(k)$ , and  $O(n \log n + kn)$  running time (ignoring dependencies on  $\varepsilon$ ). See Theorem 16. Although this result was known before, all previous constructions (including suboptimal ones), from Levcopoulos *et al.* [26] to Solomon’s work [31], as listed in Table 1.2, require intricate details. It is remarkable how effortlessly we achieve optimal  $O(k)$  degree, compared to the previous methods. (Note, however, that some of the more recent previous constructions more generally apply to spaces with bounded doubling dimension, and some also achieve good bounds on other parameters such as the total weight and the hop-diameter.) Our algorithm can be easily made dynamic, with  $O(\log n + k)$  update time. No previous results on dynamic fault-tolerant spanners were known.
- (e) **Approximate nearest neighbors.** Locality-sensitive orderings lead to a simple dynamic data structure for  $(1 + \varepsilon)$ -approximate nearest neighbor search with  $O(\log n)$  time per update/query (ignoring dependencies on  $\varepsilon$ ). While this result is not new [7], we emphasize that the query algorithm is the simplest so far – it is just a binary search in the orderings maintained.

## Computational models

The model of computation we have assumed is a unit-cost real RAM, supporting standard arithmetic operations and comparisons (but no floor function), augmented with bitwise-logical operations (bitwise-exclusive-or and bitwise-and), which are commonly available in programming languages (and in reality are cheaper than some arithmetic operations like multiplication).

If we assume that input coordinates are integers bounded by  $U$  and instead work in the word RAM model with  $(\log U)$ -bit words ( $U \geq n$ ), then our approach can actually yield *sublogarithmic* query/update time. For example, we can achieve  $O(\log \log U)$  expected time for dynamic approximate bichromatic closest pair, dynamic spanners, and dynamic ANN, by replacing binary search with van Emde Boas trees [32]. Sublogarithmic algorithms were known before for dynamic ANN [7], but ours is the first sublogarithmic result for dynamic  $(1 + \varepsilon)$ -spanners. Our results also answer the open problem of dynamic  $(1 + \varepsilon)$ -approximate bichromatic closest pair in sublogarithmic time, originally posed by Chan and Skrepetos [10].

## 2 Locality-sensitive orderings

### 2.1 Grids and orderings

► **Definition.** For a set  $X$ , consider a total order (or ordering)  $\prec$  on the elements of  $X$ . Two elements  $x, y \in X$  are **adjacent** if there is no element  $z \in X$ , such that  $x \prec z \prec y$  or  $y \prec z \prec x$ .

Given two elements  $x, y \in X$ , such that  $x \prec y$ , the **interval**  $[x, y)$  is the set  $[x, y) = \{x\} \cup \{z \in X \mid x \prec z \prec y\}$ .

The following is well known, and goes back to a work by Walecki in the 19th century [1]. We include a proof in Appendix A.1 for the sake of completeness.

► **Lemma 1.** *For  $n$  elements  $\{0, \dots, n-1\}$ , there is a set  $\mathfrak{D}$  of  $\lceil n/2 \rceil$  orderings of the elements, such that, for all  $i, j \in \{0, \dots, n-1\}$ , there exist an ordering  $\sigma \in \mathfrak{D}$  in which  $i$  and  $j$  are adjacent.*

► **Definition 2.** Consider an axis-parallel cube  $\mathcal{C} \subseteq \mathbb{R}^d$  with side length  $\ell$ . Partitioning it uniformly into a  $t \times t \times \dots \times t$  grid  $\mathbf{G}$  creates the  **$t$ -grid** of  $\mathcal{C}$ . The grid  $\mathbf{G}$  is a set of  $t^d$  identically sized cubes with side length  $\ell/t$ .

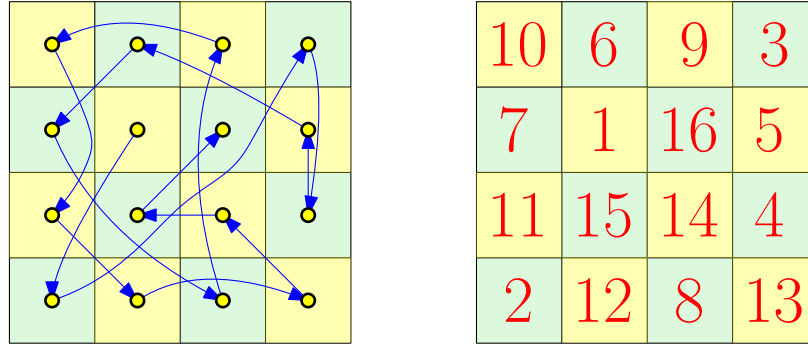
For a cube  $\square \subseteq \mathbb{R}^d$ , its **diameter** is  $\text{diam}(\square) = \text{sidelength}(\square)\sqrt{d}$ .

By Lemma 1 we obtain the following result.

► **Corollary 3.** *For a  $t$ -grid  $\mathbf{G}$  of an axis-parallel cube  $\mathcal{C} \subseteq \mathbb{R}^d$ , there is a set  $\mathfrak{D}(t, d)$  of  $O(t^d)$  orderings, such that for any  $\square_1, \square_2 \in \mathbf{G}$ , there exists an order  $\sigma \in \mathfrak{D}(t, d)$  where  $\square_1$  and  $\square_2$  are adjacent in  $\sigma$ .*

### 2.2 $\varepsilon$ -Quadtrees

► **Definition 4.** An  **$\varepsilon$ -quadtrees**  $\mathcal{T}_\varepsilon$  is a quadtree-like structure, built on a cube with side length  $\ell$ , where each cell is being partitioned into a  $(1/\varepsilon)$ -grid. The construction then continues recursively into each grid cell of interest. As such, a node in this tree has up to  $1/\varepsilon^d$  children, and a node at level  $i \geq 0$  has an associated cube of side length  $\ell\varepsilon^i$ . We call a  $1/2$ -quadtrees a regular quadtree.



■ **Figure 2.1** One ordering of a set of cells.

► **Lemma 5.** Let  $E > 0$  be an integer number,  $\varepsilon = 2^{-E}$ , and  $\mathcal{T}$  be a regular quadtree over  $[0, 2)^d$ . Then there are  $\varepsilon$ -quadtrees  $\mathcal{T}_\varepsilon^1, \dots, \mathcal{T}_\varepsilon^E$ , such that the collection of cells at each level in  $\mathcal{T}$  is contained in exactly one of these  $\varepsilon$ -quadtrees.

**Proof.** For  $i = 0, \dots, E - 1$ , construct the  $\varepsilon$ -quadtree  $\mathcal{T}_\varepsilon^i$  using the cube  $[0, 2^{E-i})^d \supseteq [0, 2)^d$  as the root. Now for  $j \in \{0, \dots, E - 1\}$ , observe that the collection of cells at levels  $j, j + E, j + 2E, \dots$ , of  $\mathcal{T}$  will also be in the quadtree  $\mathcal{T}_\varepsilon^j$ . Indeed, any node at level  $j + \ell E$  in  $\mathcal{T}$  corresponds to a cell of side length  $2^{-(j+\ell E)}$ . Now in the  $(\ell + 1)$ th level of quadtree  $\mathcal{T}_\varepsilon^j$ , this same node will have side length  $\varepsilon^{\ell+1} 2^{E-j} = 2^{-(j+\ell E)}$ . ◀

Consider an  $\varepsilon$ -quadtree  $\mathcal{T}_\varepsilon$ . Every node has up to  $1/\varepsilon^d$  children. Consider any ordering  $\sigma$  of  $\{1, \dots, 1/\varepsilon^d\}$ , and consider a DFS of  $\mathcal{T}_\varepsilon$  that always visits the children of a node in the order specified by  $\sigma$ . This induces an ordering on the points in the cube which is the root of  $\mathcal{T}_\varepsilon$ . Indeed, for any two points, imagine storing them in an  $\varepsilon$ -quadtree – this implies that the two points are each stored in their own leaf node, which contains no other point of interest. Now, independently of what other points are stored in the quadtree, this DFS traversal would visit these two points in the same order. This can be viewed as a space filling curve (which is not continuous) which maps a cube to an interval. This is a generalization of the  $\mathcal{Z}$ -order. In particular, given a point set stored in  $\mathcal{T}_\varepsilon$ , and  $\sigma$ , one can order the points according to this DFS traversal, resulting in 1-dimensional ordering of the points. We denote the resulting ordering by  $(\mathcal{T}_\varepsilon, \sigma)$ .

► **Definition 6.** Let  $\Pi$  be the set of all orderings of  $[0, 2)^d$ , induced by picking one of the  $\lg(1/\varepsilon)$  trees of Lemma 5, together with an ordering  $\sigma \in \mathcal{D}(1/\varepsilon, d)$ , as defined by Lemma 1. Each ordering in  $\Pi$  is called an  **$\varepsilon$ -ordering**.

Any two points that are lucky enough to lie in a cell of the quadtree that has diameter close to their distance, are going to be adjacent in one of the  $\varepsilon$ -orderings. Indeed, consider two points  $p, q \in [0, 1)^d$ , a parameter  $\varepsilon > 0$ , such that  $p, q$  are both contained in a cell  $\square$  of the regular quadtree  $\mathcal{T}$  with  $\|p - q\| > \varepsilon \text{diam}(\square)$ . Then, there is an  $\varepsilon$ -quadtree  $\mathcal{T}_\varepsilon$  that has  $\square$  as a node, and let  $\square_p$  and  $\square_q$  be the two children of  $\square$  in  $\mathcal{T}_\varepsilon$ , containing  $p$  and  $q$  respectively. Furthermore, there is an ordering  $\sigma \in \mathcal{D}(1/\varepsilon, d)$ , such that  $\square_p$  and  $\square_q$  are adjacent. As such, the cube  $\square_p$  (resp.,  $\square_q$ ) corresponds to an interval  $[x, x')$  (resp.,  $[x', x'')$ ) in the ordering  $(\mathcal{T}_\varepsilon, \sigma)$ , and these two intervals are adjacent.

Now consider the case when there are two points close together, but no appropriately sized quadtree cell contains both  $p$  and  $q$ . In other words, two points that are close together might get separated by nodes that are much bigger in the quadtree. This can be resolved using shifting. We need the following result of Chan [6, Lemma 3.3] – a proof is provided in Appendix A.2.



► **Lemma 7.** Consider any two points  $p, q \in [0, 1]^d$ , and let  $\mathcal{T}$  be the infinite quadtree of  $[0, 2]^d$ . For  $D = 2 \lceil d/2 \rceil$  and  $i = 0, \dots, D$ , let  $v_i = (i/(D+1), \dots, i/(D+1))$ . Then there exists an  $i \in \{0, \dots, D\}$ , such that  $p + v_i$  and  $q + v_i$  are contained in a cell of  $\mathcal{T}$  with side length  $\leq 2(D+1) \|p - q\|$ .

### 2.3 Comparing two points according to an $\varepsilon$ -ordering

We now show how to efficiently compare two points in  $P$  according to a given  $\varepsilon$ -ordering  $\sigma$  with a shift  $v_i$ . The shift can be added to the two points directly, and as such only need to worry about how to compare two points according to  $\sigma$ .

► **Observation 8.** Let  $\oplus$  denote the bitwise-exclusive-or operator. Define  $\text{msb}(a) := -\lfloor \lg a \rfloor$  to be the index of the most significant bit in the binary expansion of  $a \in (0, 2]$ . Given  $a, b \in [0, 2)$ , one can compare the  $\text{msb}$  of two numbers using the following:

(a)  $\text{msb}(a) > \text{msb}(b)$  if and only if  $a < b$  and  $a < a \oplus b$ .

(b) If  $a \oplus b \leq a \wedge b$  then  $\text{msb}(a) = \text{msb}(b)$ , where  $\wedge$  is the bitwise-and operator.

► **Lemma 9.** Let  $p = (p_1, \dots, p_d)$  and  $q = (q_1, \dots, q_d)$  be two distinct points in  $P \subseteq [0, 2]^d$  and  $\sigma \in \Pi$  be an  $\varepsilon$ -ordering over the cells of some  $\varepsilon$ -quadtree  $\mathcal{T}_\varepsilon$  storing  $P$ . Then one can determine if  $p \prec_\sigma q$  using  $O(d \log(1/\varepsilon))$  bitwise-logical operations.

**Proof.** Recall  $\varepsilon$  is a power of two and  $E = \lg(1/\varepsilon)$ . In order to compare  $p$  and  $q$ , for  $i = 1, \dots, d$ , compute  $a_i = p_i \oplus q_i$ . Find the index  $i'$  such that  $\text{msb}(a_{i'}) \leq \text{msb}(a_i)$  for all  $i$ , using  $O(d)$  comparisons. Given  $p_{i'}$  and  $q_{i'}$ , our first goal is to determine the place in which  $p_{i'}$  and  $q_{i'}$  first differ in their binary representation. Note that because  $\varepsilon$  is a power of two, each bit in the base  $1/\varepsilon$  expansion of  $p_{i'}$  corresponds to a block of  $E$  bits in the binary expansion of  $p_{i'}$ .

First we find the position within the block of size  $E$  where  $p_{i'}$  and  $q_{i'}$  differ in their binary expansion. For  $j = 1, \dots, E$ , let  $b_j = 2^{E-j}/(2^E - 1) \in (0, 1)$  be the number whose binary expansion has a 1 in positions  $j, j + E, j + 2E, \dots$ , and 0 everywhere else. For  $j = 1, \dots, E$ , compute  $b_j \wedge a_{i'}$  and check if  $\text{msb}(b_j) = \text{msb}(b_j \wedge a_{i'})$ . Once we find a  $j$  such that  $\text{msb}(b_j) = \text{msb}(b_j \wedge a_{i'})$ , stop. We know that that  $p_{i'}$  and  $q_{i'}$  first differ in the  $j$ th position inside some block.

It remains to extract the  $E$  bits from each block in  $p_1, \dots, p_d$ . For  $i = 1, \dots, d$ , let  $B_i \in \{0, 1\}^E$  be the bits inside the block associated with  $p_i$ . For  $k = 1, \dots, E$ , set  $B_{i,k} = \mathbb{1}[\text{msb}(2^{j-k} a_{i'}) = \text{msb}((2^{j-k} a_{i'}) \wedge p_i)]$  (where  $\mathbb{1}[\cdot]$  is the indicator function). By repeating a similar process for all  $q_1, \dots, q_d$ , we obtain the coordinates of the cells in which  $p$  and  $q$  differ. We can then consult  $\sigma$  to determine whether or not  $p \prec_\sigma q$ .

This implies that  $p$  and  $q$  can be compared using  $O(d \log(1/\varepsilon))$  operations by Observation 8. ◀

#### Remark

In the word RAM model for integer input, the extra  $\log(1/\varepsilon)$  factor in the above time bound can be eliminated:  $\text{msb}$  can be explicitly computed in  $O(1)$  time by a complicated algorithm of Fredman and Willard [15]; this allows us to directly jump to the right block of each coordinate and extract the relevant bits.

## 2.4 The result

► **Theorem 10.** *For a parameter  $\varepsilon \in (0, 1)$ , there is a set  $\Pi^+$  of  $O((1/\varepsilon^d) \log(1/\varepsilon))$  orderings of  $[0, 1]^d$ , such that for any two points  $p, q \in [0, 1]^d$  there is an ordering  $\sigma \in \Pi^+$  defined over  $[0, 1]^d$ , such that for any point  $u$  with  $p \prec_\sigma u \prec_\sigma q$  it holds that either  $\|p - u\| \leq \varepsilon \|p - q\|$  or  $\|q - u\| \leq \varepsilon \|p - q\|$ .*

*Furthermore, given such a ordering  $\sigma$ , and two points  $p, q$ , one can compute their ordering, according to  $\sigma$ , using  $O(d \log(1/\varepsilon))$  arithmetic and bitwise-logical operations.*

**Proof.** Let  $\Pi^+$  be the set of all ordering defined by picking an ordering from  $\Pi$ , as defined by Definition 6 using the parameter  $\varepsilon$ , together with a shift from Lemma 7.

Consider any two points  $p, q \in [0, 1]^d$ . By Lemma 7 there is a shift for which the two points fall into a quadtree cell  $\square$  with side length at most  $2(D + 1) \|p - q\|$ . Next, there is an  $\varepsilon$ -quadtree  $\mathcal{T}_\varepsilon$  that contains  $\square$ , and the two children that correspond to two cells  $\square_p$  and  $\square_q$  with side length at most  $2(D + 1)\varepsilon \|p - q\|$ , which readily implies that the diameter of these cells is at most  $2(D + 1)\sqrt{d}\varepsilon \|p - q\|$ . Furthermore, there is an  $\varepsilon$ -ordering in  $\Pi$  such that all the points of  $\square_p$  are adjacent to all the points  $\square_q$  in this ordering. This implies the desired claim, after adjusting  $\varepsilon$  by a factor of  $2(D + 1)\sqrt{d}$  (and rounding to a power of 2). ◀

From now on, we refer to the set of orderings  $\Pi^+$  in the above Theorem as locality-sensitive orderings. We remark that by the readjustment of  $\varepsilon$  in the final step of the proof, the number of locality-sensitive orderings when including the factors involving  $d$  is  $O(d^{3d/2}(1/\varepsilon^d) \log(1/\varepsilon))$ .

### 2.4.1 Discussion

#### Connection to locality-sensitive hashing

Let  $P$  be a set of  $n$  points in  $\{0, 1\}^d$ . Consider the decision version of the  $(1 + \varepsilon)$ -approximate nearest neighbor problem. Specifically, for a pre-specified radius  $r$  and any given query point  $q$ , we would like to efficiently decide whether or not there exists a point  $p \in P$  such that  $\|q - p\|_1 \leq (1 + \varepsilon)r$  or conclude that all points in  $P$  are at least distance  $r$  from  $q$ . The locality-sensitive hashing (LSH) technique [23] can be used to create a data structure supporting these types of decision queries in time  $O(dn^{1/(1+\varepsilon)} \log n)$  time (which is correct with high probability) and using total space  $O(dn^{1+1/(1+\varepsilon)} \log n)$ . Similar results also hold in Euclidean space.

At a high level, LSH works as follows. Start by choosing  $k := k(\varepsilon, r, n)$  indices in  $[d]$  at random (with replacement). Let  $R$  denote the resulting multiset of coordinates. For each point  $p \in P$ , let  $p_R$  be the projection  $p$  onto these coordinates of  $R$ . We can group the points of  $P$  into buckets, where each bucket contains points with the same projection. Given a query point  $q$ , we check if any of the points in the same bucket as  $q_R$  is at distance at most  $(1 + \varepsilon)r$  from  $q$ . This construction can also be repeated a sufficient number of times in order to guarantee success with high probability.

The idea of bucketing can also be viewed as an implicit ordering on the randomly projected point set by ordering points lexicographically according to the  $k$  coordinates. In this sense, the query algorithm can be viewed as locating  $q$  within each of the orderings, and comparing  $q$  to similar points nearby in each ordering. From this perspective, every locality-sensitive ordering can be viewed as a LSH scheme. Indeed, for a given query point  $q$ , the approximate nearest neighbor to  $q$  can be found by inspecting the elements adjacent to  $q$  in each of the locality-sensitive orderings and returning the closest point to  $q$  found (see Theorem 17).

Of course, the main difference between the two schemes is that for every fixed  $\varepsilon$ , the number of “orderings” in a LSH scheme is polynomial in both  $d$  and  $n$ . While for locality-sensitive orderings, the number of orderings remain exponential in  $d$ . This trade-off is required, as locality-sensitive orderings guarantee a much stronger property than that of a LSH scheme.

### Extension of locality-sensitive orderings to other norms in Euclidean space

The result of Theorem 10 also holds for any  $L_p$  norm. The key change that is needed is in the proof of Lemma 7: For any two points  $s, t \in [0, 1]^d$ , there exists a shift  $v$  such that  $s + v$  and  $t + v$  are contained in a quadtree cell of side length at most  $2(D + 1) \|s - t\|_p$ . This extension follows easily from the proof of the Lemma, see Appendix A.2. Theorem 10 then follows by adjusting  $\varepsilon$  by a factor of  $2(D + 1)d^{1/p}$  in the last step, implying that the number of orderings will be  $O(d^{d(1+1/p)}(1/\varepsilon^d) \log(1/\varepsilon))$ .

### Extension of locality-sensitive orderings for doubling metrics

An abstraction of low-dimensional Euclidean space, is a metric space with (low) doubling dimension. Formally, a metric space  $(\mathcal{M}, d)$  has *doubling dimension*  $\lambda$  if any ball of  $\mathcal{M}$  of radius  $r$  can be covered by at most  $2^\lambda$  balls of half the radius (i.e.,  $r/2$ ). It is known that  $\mathbb{R}^d$  has doubling dimension  $O(d)$  [33]. We point out that locality-sensitive orderings still exist in this case, but they are less constructive in nature, since one needs to be provided with all the points of interest in advance.

For a point set  $P \subseteq \mathcal{M}$ , the analogue of a quadtree for a metric space is a net tree [20]. Being somewhat imprecise, a net tree can be constructed as follows: The root node corresponds to the point set  $P \subseteq \mathcal{M}$ . Compute a randomized partition of  $P$  of diameter  $1/2$  (assume  $P$  has diameter one), and for each cluster in the partition, create an associated node and hang it on the root. The tree is computed recursively in this manner, at each level  $i$  computing a random partition of diameter  $2^{-i}$ . The leaves of the tree are points of  $P$ .

As with quadtrees, it is possible during this randomized construction for two nearby points to be placed in different clusters and be separated further down the tree. If  $\ell = d(p, q)$  for two points  $p, q \in P$ , then the probability that  $p$  and  $q$  lie in different clusters of diameter  $r = 2^{-i}$  in the randomized partition is at most  $O((\ell/r) \log n)$  [13]. In particular, for  $r \approx 1/(\ell \log n)$ , the probability  $p$  and  $q$  are separated is at most a constant. If we want even this property to hold with high probability for all pairs of points, one needs to construct  $O(\log n)$  (randomly constructed) net trees of  $P$ . (This corresponds to randomly shifting a quadtree  $O(\log n)$  times in the Euclidean setting.)

Given such a net tree  $T$ , each node has  $I = 2^{O(\lambda)}$  children. We can arbitrarily and explicitly number the children of each node by a distinct label from  $[I]$ . One can define an ordering of such a tree as we did in the Euclidean case, except that the gap (in diameter) between a node and its children is  $O(\varepsilon/\log n)$  instead of  $\varepsilon$ . Repeating our scheme in the Euclidean case, this implies that one would expect to require  $(\varepsilon^{-1} \log n)^{O(\lambda)}$  orderings of  $P$ .

This requires having all the points of  $P$  in advance, which is a strong assumption for a dynamic data structure (like the applications we show next). For example, Gottlieb and Roditty [17] show how to maintain dynamic spanners in a doubling metric, but only assuming that after a point has been deleted from  $P$ , the distance between the deleted point and a point currently in  $P$  can still be computed in constant time.

### 3 Applications

#### 3.1 Bichromatic closest pair

Given an ordering  $\sigma \in \Pi^+$ , and two finite sets of points  $R, B$  in  $\mathbb{R}^d$ , let  $\mathcal{Z} = \mathcal{Z}(\sigma, R, B)$  be the set of all pairs of points in  $R \times B$  that are adjacent in the ordering of  $R \cup B$  according to  $\sigma$ . Observe that inserting or deleting a single point from these two sets changes the content of  $\mathcal{Z}$  by a constant number of pairs. Furthermore, a point participates in at most two pairs.

► **Lemma 11.** *Let  $R$  and  $B$  be two sets of points in  $[0, 1]^d$ , and let  $\varepsilon \in (0, 1)$  be a parameter. Let  $\sigma \in \Pi^+$  be a locality-sensitive ordering (see Theorem 10). Then, one can maintain the set  $\mathcal{Z} = \mathcal{Z}(\sigma, R, B)$  under insertions and deletions to  $R$  and  $B$ . In addition, one can maintain the closest pair in  $\mathcal{Z}$  (under the Euclidean metric). Each update takes  $O(d \log(n) \log(1/\varepsilon))$  time, where  $n$  is the total size of  $R$  and  $B$  during the update operation.*

**Proof.** Maintain two balanced binary search trees  $T_R$  and  $T_B$  storing the points in  $R$  and  $B$ , respectively, according to the order  $\sigma$ . Insertion, deletion, predecessor query and successor query can be implemented in  $O(d \log(1/\varepsilon) \log n)$  time (since any query requires  $O(\log n)$  comparisons costing each  $O(d \log(1/\varepsilon))$  time by Lemma 9). We also maintain a min-heap of the pairs in  $\mathcal{Z}$  sorted according to the Euclidean distance. The minimum is the desired closest pair. Notice that a single point can participate in at most two pairs in  $\mathcal{Z}$ .

We now explain how to handle updates. Given a newly inserted point  $r$  (say a red point that belongs to  $R$ ), we compute its (potential) pairs it participates in, by computing its successor  $r'$  in  $R$ , and its successor  $b'$  in  $B$ . If  $r \prec_\sigma b' \prec_\sigma r'$  then the new pair  $rb'$  should be added to  $\mathcal{Z}$ . The pair before  $r$  in the ordering that might use  $r$  is computed in a similar fashion. In addition, we recompute the predecessor and successor of  $r$  in  $R$ , and we recompute the pairs they might participate in (deleting potentially old pairs that are no longer valid).

Deletion is handled in a similar fashion – all points included in pairs with the deleted point recompute their pairs. In addition, the successor and predecessor (of the same color) need to recompute their pairs. This all requires a constant number of queries in the two trees, and thus takes the running time as stated. ◀

► **Theorem 12.** *Let  $R$  and  $B$  be two sets of points in  $[0, 1]^d$ , and let  $\varepsilon \in (0, 1)$  be a parameter. Then one can maintain a  $(1 + \varepsilon)$ -approximation to the bichromatic closest pair in  $R \times B$  under updates (i.e., insertions and deletions) in  $O(\log(n) \log^2(1/\varepsilon)/\varepsilon^d)$  time per operation, where  $n$  is the total number of points in the two sets. The data structure uses  $O(n \log(1/\varepsilon)/\varepsilon^d)$  space, and at all times maintains a pair of points  $r \in R$ ,  $b \in B$ , such that  $\|r - b\| \leq (1 + \varepsilon)d(R, B)$ , where  $d(R, B) = \min_{r \in R, b \in B} \|r - b\|$ .*

**Proof.** We maintain the data structure of Lemma 11 for all the locality-sensitive orderings of Theorem 10. We might as well maintain all the good pairs for these data structures together in one global min-heap. We claim that the minimum length pair in this heap is the desired approximation.

To see that, consider the bichromatic closest pair  $r \in R$  and  $b \in B$ . By Theorem 10 there is a locality-sensitive ordering  $\sigma$ , such that the interval  $I$  in the ordering between  $r$  and  $b$  contains points that are in distance at most  $\ell = \varepsilon \|r - b\|$  from either  $r$  or  $b$ . In particular, let  $P_r$  (resp.,  $P_b$ ) be all the points in  $I$  in distance at most  $\ell$  from  $r$  (resp.,  $b$ ). Observe that  $P_r \subseteq R$ , as otherwise, there would be a bichromatic pair in  $P_r$ , and since the diameter of this set is at most  $\ell$ , this would imply that  $(r, b)$  is not the closest bichromatic pair – a contradiction. Similarly,  $P_b \subseteq B$ . As such, there must be two points  $b' \in B$  and  $r' \in R$ , that are consecutive in  $\sigma$ , and this is one of the pairs considered by the algorithm (as it is stored

in the min-heap). In particular, by the triangle inequality, we have

$$\|r' - b'\| \leq \|r' - r\| + \|r - b\| + \|b - b'\| \leq 2\ell + \|r - b\| \leq (1 + 2\varepsilon) \|r - b\|.$$

The theorem follows after adjusting  $\varepsilon$  by a factor of 2.  $\blacktriangleleft$

### Remark

In the word RAM model, for integer input in  $\{1, \dots, U\}^d$ , the update time can be improved to  $O((\log \log U) \log^2(1/\varepsilon)/\varepsilon^d)$  expected, by using van Emde Boas trees [32] in place of the binary search trees (and the min-heaps as well). With standard word operations, we may not be able to explicitly map each point to an integer in one dimension following each locality-sensitive ordering, but we can still simulate van Emde Boas trees on the input as if the mapping has been applied. Each recursive call in the van Emde Boas recursion focuses on a specific block of bits of each input coordinate value (after shifting); we can extract these blocks, and perform the needed hashing operations on the concatenation of these blocks over the  $d$  coordinates of each point.

## 3.2 Dynamic spanners

► **Definition 13.** For a set of  $n$  points  $P$  in  $\mathbb{R}^d$  and a parameter  $t \geq 1$ , a  $t$ -**spanner** of  $P$  is an undirected graph  $G = (P, E)$  such that for all  $p, q \in P$ ,

$$\|p - q\| \leq d_G(p, q) \leq t\|p - q\|,$$

where  $d_G(p, q)$  is the length of the shortest path from  $p$  to  $q$  in  $G$  using the edge set  $E$ .

Using a small modification of the results in the previous section, we easily obtain a dynamic  $(1 + \varepsilon)$ -spanner. Note that there is nothing special about how the data structure in Theorem 12 deals with the bichromatic point set. If the point set is monochromatic, modifying the data structure in Lemma 11 to account for the closest monochromatic pair of points leads to a data structure with the same bounds and maintains the  $(1 + \varepsilon)$ -approximate closest pair.

The construction of the spanner is very simple: Given  $P$  and  $\varepsilon \in (0, 1)$ , maintain orderings of the points specified by  $\Pi^+$  (see Theorem 10). For each  $\sigma \in \Pi^+$ , let  $E_\sigma$  be the edge set consisting of edges connecting two consecutive points according to  $\sigma$ , with weight equal to their Euclidean distance. Thus  $|E_\sigma| = n - 1$ . Our spanner  $G = (P, E)$  then consists of the edge set  $E = \bigcup_{\sigma \in \Pi^+} E_\sigma$ .

► **Theorem 14.** *Let  $P$  be a set of  $n$  points in  $[0, 1]^d$  and  $\varepsilon \in (0, 1)$ . One can compute a  $(1 + \varepsilon)$ -spanner  $G$  of  $P$  with  $O(n \log(1/\varepsilon)/\varepsilon^d)$  edges, where every vertex has degree  $O(\log(1/\varepsilon)/\varepsilon^d)$ . Furthermore, insertions and deletions can be performed in  $O(\log(n) \log^2(1/\varepsilon)/\varepsilon^d)$  time, with at most  $O(\log(1/\varepsilon)/\varepsilon^d)$  edges being deleted or inserted into the spanner.*

**Proof.** The construction is described above. The same analysis as in the proof of Theorem 12 implies the number of edges in  $G$  and the update time.

It remains to prove that  $G$  is a spanner. By Theorem 10, for any pair of points  $s, t \in P$ , there is a locality-sensitive ordering  $\sigma \in \Pi^+$ , such that the  $\sigma$ -interval  $[s, t)$  contains only points that are in distance at most  $\varepsilon \|s - t\|$  from either  $s$  or  $t$ . In particular, there must be two points in  $s', t' \in P$  that are adjacent in  $\sigma$ , such that one of them, say  $s'$  (resp.,  $t'$ ) is in distance at most  $\varepsilon \|s - t\|$  from  $s$  (resp.,  $t$ ). As such, the edge  $s't'$  exists in the graph being maintained.

This property is already enough to imply that this graph is a  $(1 + c\varepsilon)$ -spanner for a sufficiently large constant  $c$  – this follows by an induction on the distances between the points (specifically, in the above, we apply the induction hypothesis on the pairs  $s, s'$  and  $t, t'$ ). We omit the easy but somewhat tedious argument – see [4] or [19, Theorem 3.12] for details. The theorem follows after adjusting  $\varepsilon$  by a factor of  $c$ . ◀

### 3.2.1 Static and dynamic vertex-fault-tolerant spanners

► **Definition 15.** For a set of  $n$  points  $P$  in  $\mathbb{R}^d$  and a parameter  $t \geq 1$ , a  $k$ -**vertex-fault-tolerant  $t$ -spanner** of  $P$ , denoted by  $(k, t)$ -VF $T$ S, is a graph  $G = (P, E)$  such that

- (i)  $G$  is a  $t$ -spanner (see Definition 13), and
- (ii) For any  $P' \subseteq P$  of size at most  $k$ , the graph  $G \setminus P'$  is a  $t$ -spanner for  $P \setminus P'$ .

A  $(k, 1 + \varepsilon)$ -VF $T$ S can be obtained by modifying the construction of the  $(1 + \varepsilon)$ -spanner in Section 3.2. Construct a set of locality-sensitive orderings  $\Pi^+$ . For each  $\sigma \in \Pi^+$  and each  $p \in P$ , connect  $p$  to its  $k + 1$  successors and  $k + 1$  predecessors according to  $\sigma$  with edge weights equal to the Euclidean distances. Thus each ordering maintains  $O(nk)$  edges and there are  $O(|\Pi^+|kn) = O(kn \log(1/\varepsilon)/\varepsilon^d)$  edges overall. We now prove that this graph  $G$  is in fact a  $(k, 1 + \varepsilon)$ -VF $T$ S.

► **Theorem 16.** *Let  $P$  be a set of  $n$  points in  $[0, 1]^d$  and  $\varepsilon \in (0, 1)$ . One can compute a  $k$ -vertex-fault-tolerant  $(1 + \varepsilon)$ -spanner  $G$  for  $P$  in time  $O((n \log n \log(1/\varepsilon) + kn) \log(1/\varepsilon)/\varepsilon^d)$ . The number of edges is  $O(kn \log(1/\varepsilon)/\varepsilon^d)$  and the maximum degree is bounded by  $O(k \log(1/\varepsilon)/\varepsilon^d)$ .*

*Furthermore, one can maintain the  $k$ -vertex-fault-tolerant  $(1 + \varepsilon)$ -spanner  $G$  under insertions and deletions in  $O((\log n \log(1/\varepsilon) + k) \log(1/\varepsilon)/\varepsilon^d)$  time per operation.*

**Proof.** The construction algorithm, number of edges, and maximum degree follows from the discussion above. So, consider deleting a set  $P' \subseteq P$  of size at most  $k$  from  $G$ . Consider an ordering  $\sigma \in \Pi^+$  with the points  $P'$  removed. By the construction of  $G$ , all the pairs of points of  $P \setminus P'$  that are (now) adjacent in  $\sigma$  remain connected by an edge in  $G \setminus P'$ . The argument of Theorem 14 implies that the remaining graph is spanner. We conclude that that  $G \setminus P'$  is a  $(1 + \varepsilon)$ -spanner for  $P \setminus P'$ .

As for the time taken to handle insertions and deletions, one simply maintains the orderings of the points using balanced search trees. After an insertion to one of the orderings in  $O(\log n \log(1/\varepsilon))$  time,  $O(k)$  edges have to be added and deleted. Therefore insertions take  $O((\log n \log(1/\varepsilon) + k) |\Pi^+|) = O((\log n \log(1/\varepsilon) + k) \log(1/\varepsilon)/\varepsilon^d)$  time total. Deletions are handled similarly.

The total construction time follows by inserting each of the points into the dynamic data structure. ◀

### 3.3 Dynamic approximate nearest neighbors

Another application of the same data structure in Theorem 12 is supporting  $(1 + \varepsilon)$ -approximate nearest neighbor queries. In this scenario, the data structure must support insertions and deletions of points and the following queries: given a point  $q$ , return a point  $t \in P$  such that  $\|q - t\| \leq (1 + \varepsilon) \min_{p \in P} \|q - p\|$ .

► **Theorem 17.** *Let  $P$  be a set of  $n$  points in  $[0, 1]^d$ . For a given  $\varepsilon > 0$ , one can build a data structure using  $O(n \log(1/\varepsilon)/\varepsilon^d)$  space, that supports insertion and deletion in time  $O(\log(n) \log^2(1/\varepsilon)/\varepsilon^d)$ . Furthermore, given a query point  $q \in [0, 1]^d$ , the data structure returns a  $(1 + \varepsilon)$ -approximate nearest neighbor in  $P$  in  $O(\varepsilon^{-d} \log^2(1/\varepsilon) \log(n))$  time.*

**Proof.** Maintain the data structure of Lemma 11 for all locality-sensitive orderings of Theorem 10, with one difference: Since the input is monochromatic, the data structures for each ordering store distances between all consecutive pairs. The space and update time bounds easily follow by the same analysis.

Given a query point  $q \in [0, 1]^d$ , for each of the orderings the algorithm inspects the predecessor and successor to  $q$ . The algorithm returns the closest point to  $q$  encountered. We claim that the returned point  $p$  is the desired approximate nearest neighbor.

Let  $p^* \in P$  be the nearest neighbor to  $q$  and  $\ell = \|q - p^*\|$ . By Theorem 10, there is a locality-sensitive ordering  $\sigma \in \Pi^+$  such that the  $\sigma$ -interval  $I = [p^*, q)$  contains points that are of distance at most  $\varepsilon\ell$  from  $p^*$  or  $q$  (and this interval contains at least one point of  $P$ , namely,  $p^*$ ). Note that no point of  $P$  can be at distance less than  $\varepsilon\ell$  to  $q$ . Thus, the point  $p \in P$  adjacent to  $q$  in  $I$  is of distance at most  $\varepsilon\ell$  from  $p^*$ . Therefore, for such a point  $p$ , we have  $\|p - q\| \leq \|p - p^*\| + \|p^* - q\| \leq (1 + \varepsilon)\ell$ .

The final query time follows from the time taken for these predecessor and successor queries, as in the proof of Lemma 11. ◀

---

## References

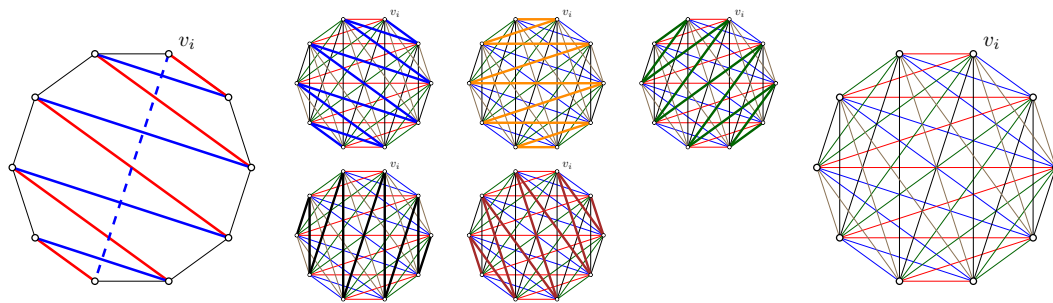
- 1 Brian Alspach. The wonderful Walecki construction. *Bull. Inst. Combin. Appl.*, 52:7–20, 2008.
- 2 Sanjeev Arora. Polynomial time approximation schemes for Euclidean TSP and other geometric problems. *J. Assoc. Comput. Mach.*, 45(5):753–782, September 1998. URL: <http://www.cs.princeton.edu/~arora/pubs/tsp.ps>.
- 3 Marshall W. Bern. Approximate Closest-Point Queries in High Dimensions. *Inform. Process. Lett.*, 45(2):95–99, 1993. doi:10.1016/0020-0190(93)90222-U.
- 4 Paul B. Callahan and S. Rao Kosaraju. Faster Algorithms for Some Geometric Graph Problems in Higher Dimensions. In Vijaya Ramachandran, editor, *Proc. 4th ACM-SIAM Sympos. Discrete Alg. (SODA)*, pages 291–300. ACM/SIAM, 1993. URL: <http://dl.acm.org/citation.cfm?id=313559.313777>.
- 5 T.-H. Hubert Chan, Mingfei Li, Li Ning, and Shay Solomon. New Doubling Spanners: Better and Simpler. *SIAM J. Comput.*, 44(1):37–53, 2015. doi:10.1137/130930984.
- 6 Timothy M. Chan. Approximate nearest neighbor queries revisited. *Discrete Comput. Geom.*, 20(3):359–373, 1998. doi:10.1007/PL00009390.
- 7 Timothy M. Chan. Closest-point problems simplified on the RAM. In *Proc. 13th ACM-SIAM Sympos. Discrete Alg. (SODA)*, pages 472–473. SIAM, 2002. URL: <http://dl.acm.org/citation.cfm?id=545381.545444>.
- 8 Timothy M. Chan. A minimalist’s implementation of an approximate nearest neighbor algorithm in fixed dimensions, 2006. URL: <http://tmc.web.engr.illinois.edu/sss.ps>.
- 9 Timothy M. Chan. Well-separated pair decomposition in linear time? *Inform. Process. Lett.*, 107(5):138–141, 2008. doi:10.1016/j.ipl.2008.02.008.
- 10 Timothy M. Chan and Dimitrios Skrepetos. Dynamic data structures for approximate Hausdorff distance in the word RAM. *Comput. Geom. Theory Appl.*, 60:37–44, 2017. doi:10.1016/j.comgeo.2016.08.002.
- 11 Artur Czumaj and Hairong Zhao. Fault-Tolerant Geometric Spanners. *Discrete Comput. Geom.*, 32(2):207–230, 2004. URL: <http://www.springerlink.com/index/10.1007/s00454-004-1121-7>.
- 12 David Eppstein. Dynamic Euclidean minimum spanning trees and extrema of binary functions. *Discrete Comput. Geom.*, 13:111–122, 1995. URL: <http://www.ics.uci.edu/~eppstein/pubs/Epp-DCG-95.pdf>.



- 13 Jittat Fakcharoenphol, Satish Rao, and Kunal Talwar. A tight bound on approximating arbitrary metrics by tree metrics. *J. Comput. Sys. Sci.*, 69(3):485–497, 2004. doi:10.1016/j.jcss.2004.04.011.
- 14 Uriel Feige and Robert Krauthgamer. Stereoscopic families of permutations, and their applications. In *Proc. 5th Israel Symp. Theo. Comput. and Systems (ISTCS)*, pages 85–95. IEEE Computer Society, 1997. doi:10.1109/ISTCS.1997.595160.
- 15 Michael L. Fredman and Dan E. Willard. Surpassing the Information Theoretic Bound with Fusion Trees. *J. Comput. Sys. Sci.*, 47(3):424–436, 1993. doi:10.1016/0022-0000(93)90040-4.
- 16 Irene Gargantini. An Effective Way to Represent Quadrees. *Commun. ACM*, 25(12):905–910, 1982. doi:10.1145/358728.358741.
- 17 Lee-Ad Gottlieb and Liam Roditty. An Optimal Dynamic Spanner for Doubling Metric Spaces. In *Proc. 16th Annu. Euro. Sympos. Alg. (ESA)*, pages 478–489, 2008. doi:10.1007/978-3-540-87744-8\_40.
- 18 Lee-Ad Gottlieb and Liam Roditty. Improved algorithms for fully dynamic geometric spanners and geometric routing. In *Proc. 19th ACM-SIAM Sympos. Discrete Alg. (SODA)*, pages 591–600, 2008. URL: <http://dl.acm.org/citation.cfm?id=1347082.1347148>.
- 19 Sariel Har-Peled. *Geometric Approximation Algorithms*, volume 173 of *Math. Surveys & Monographs*. Amer. Math. Soc., Boston, MA, USA, 2011. doi:10.1090/surv/173.
- 20 Sariel Har-Peled and Manor Mendel. Fast Construction of Nets in Low Dimensional Metrics, and Their Applications. *SIAM J. Comput.*, 35(5):1148–1184, 2006. doi:10.1137/S0097539704446281.
- 21 Dorit S. Hochbaum and Wolfgang Maass. Approximation Schemes for Covering and Packing Problems in Image Processing and VLSI. *J. Assoc. Comput. Mach.*, 32(1):130–136, 1985. doi:10.1145/2455.214106.
- 22 Jacob Holm, Eva Rotenberg, and Christian Wulff-Nilsen. Faster Fully-Dynamic Minimum Spanning Forest. In Nikhil Bansal and Irene Finocchi, editors, *Proc. 23rd Annu. Euro. Sympos. Alg. (ESA)*, volume 9294 of *Lect. Notes in Comp. Sci.*, pages 742–753. Springer, 2015. doi:10.1007/978-3-662-48350-3\_62.
- 23 Piotr Indyk and Rajeev Motwani. Approximate Nearest Neighbors: Towards Removing the Curse of Dimensionality. In *Proc. 30th ACM Sympos. Theory Comput. (STOC)*, pages 604–613, 1998. doi:10.1145/276698.276876.
- 24 Ibrahim Kamel and Christos Faloutsos. On Packing  $R$ -trees. In Bharat K. Bhargava, Timothy W. Finin, and Yelena Yesha, editors, *Proc. 2nd Intl. Conf. Info. Knowl. Mang.*, pages 490–499. ACM, 1993. doi:10.1145/170088.170403.
- 25 Sanjiv Kapoor and Xiang-Yang Li. Efficient Construction of Spanners in  $d$ -Dimensions. *CoRR*, abs/1303.7217, 2013. URL: <http://arxiv.org/abs/1303.7217>.
- 26 Christos Levcopoulos, Giri Narasimhan, and Michiel H. M. Smid. Efficient Algorithms for Constructing Fault-Tolerant Geometric Spanners. In Jeffrey Scott Vitter, editor, *Proc. 30th ACM Sympos. Theory Comput. (STOC)*, pages 186–195. ACM, 1998. doi:10.1145/276698.276734.
- 27 Swanwa Liao, Mario A. López, and Scott T. Leutenegger. High Dimensional Similarity Search With Space Filling Curves. In *Proc. 17th Int. Conf. on Data Eng. (ICDE)*, pages 615–622, 2001. doi:10.1109/ICDE.2001.914876.
- 28 Tamás Lukovszki. New Results of Fault Tolerant Geometric Spanners. In Frank K. H. A. Dehne, Arvind Gupta, Jörg-Rüdiger Sack, and Roberto Tamassia, editors, *Proc. 6th Workshop Alg. Data Struct. (WADS)*, volume 1663 of *Lect. Notes in Comp. Sci.*, pages 193–204. Springer, 1999. doi:10.1007/3-540-48447-7\_20.
- 29 Tamás Lukovszki. *New results on geometric spanners and their applications*. PhD thesis, University of Paderborn, Germany, 1999. URL: <http://d-nb.info/958256713>.

- 30 Liam Roditty. Fully Dynamic Geometric Spanners. *Algorithmica*, 62(3-4):1073–1087, 2012. doi:10.1007/s00453-011-9504-7.
- 31 Shay Solomon. From hierarchical partitions to hierarchical covers: Optimal fault-tolerant spanners for doubling metrics. In David B. Shmoys, editor, *Proc. 46th ACM Sympos. Theory Comput. (STOC)*, pages 363–372. ACM, 2014. doi:10.1145/2591796.2591864.
- 32 Peter van Emde Boas. Preserving Order in a Forest in Less Than Logarithmic Time and Linear Space. *Inf. Process. Lett.*, 6(3):80–82, 1977. doi:10.1016/0020-0190(77)90031-X.
- 33 Jean-Louis Verger-Gaugry. Covering a Ball with Smaller Equal Balls in  $\mathbb{R}^n$ . *Discrete Comput. Geom.*, 33(1):143–155, 2005. doi:10.1007/s00454-004-2916-2.

## A Proofs



■ **Figure A.1** For  $n$  even, a decomposition of  $K_n$  into  $n/2$  Hamiltonian paths.

### A.1 Proof of Lemma 1

► **Lemma 1 (Restated).** For  $n$  elements  $\{0, \dots, n-1\}$ , there is a set  $\mathfrak{D}$  of  $\lceil n/2 \rceil$  orderings of the elements, such that, for all  $i, j \in \{0, \dots, n-1\}$ , there exist an ordering  $\sigma \in \mathfrak{D}$  in which  $i$  and  $j$  are adjacent.

**Proof.** As mentioned earlier this is well known, see [1]. Assume  $n$  is even, and consider the clique  $K_n$ , with its vertices  $v_0, \dots, v_{n-1}$ . The edges of this clique can be covered by  $n/2$  Hamiltonian paths that are edge disjoint. Tracing one of these path gives rise to one ordering, and doing this for all paths results with orderings with the desired property, since edge  $v_i v_j$  is adjacent in one of these paths.

To get this cover, draw  $K_n$  by using the vertices of an  $n$ -regular polygon, and draw all the edges of  $K_n$  as straight segments. For every edge  $v_i v_{i+1}$  of  $K_n$  there are exactly  $n/2$  parallel edges with this slope (which form a matching). Let  $M_i$  denote this matching. Similarly, for the vertex  $v_i$ , consider the segment  $v_i v_{i+n/2}$  (indices are here modulo  $n$ ), and the family of segments (i.e., edges) of  $K_n$  that are orthogonal to this segment. This family is also a matching  $M'_i$  of size  $n/2 - 1$ . Observe that  $\sigma_i = M_i \cup M'_i$  form a Hamiltonian path, as shown in Figure A.1. Since the slopes of the segments in  $M_i$  and  $M'_i$  are unique, for  $i = 0, \dots, n/2 - 1$ , it follows that  $\sigma_0, \dots, \sigma_{n/2-1}$  are an edge disjoint cover of all the edges of  $K_n$  by  $n/2$  Hamiltonian paths.

If  $n$  is odd, use the above construction for  $n + 1$ , and delete the redundant symbol from the computed orderings. ◀

## A.2 Proof of 7 (shifting)

For two positive real numbers  $x$  and  $y$ , let

$$x \% y = x - y \lfloor x/y \rfloor.$$

The basic idea behind shifting is that one can pick a set of values that look the “same” in all resolutions.

► **Lemma 18.** *Let  $n > 1$  be a positive odd integer, and consider the set  $X = \{i/n \mid i = 0, \dots, n-1\}$ . Then, for any  $\alpha = 2^{-\ell}$ , where  $\ell \geq 0$  is integer, we have that  $X \% \alpha = \{i/n \% \alpha \mid i = 0, \dots, n-1\}$  is equal to the set  $\alpha X = \{\alpha i/n \mid i = 0, \dots, n-1\}$ .*

**Proof.** The proof is by induction. For  $\ell = 0$  the claim clearly holds. Next, assume the claim holds for some  $i \geq 0$ , and consider  $\ell = i + 1$ . Setting  $m = (n-1)/2$  and  $\Delta = 2^{-i}/n$ , we have by induction (and rearrangement) that

$$\begin{aligned} X \% 2^{-i} &= 2^{-i} X = \{0, \Delta, \dots, 2m\Delta\} \\ &= \{0, (m+1)\Delta, \Delta, (m+2)\Delta, 2\Delta, \dots, (m+m)\Delta, m\Delta\}. \end{aligned}$$

Settings  $\delta = \Delta/2 = 2^{-i-1}/n$ , and observing that  $(2m+1)\delta = n\delta = 2^{-i-1}$ , we have

$$\begin{aligned} X \% 2^{-i-1} &= (X \% 2^{-i}) \% 2^{-i-1} \\ &= \{0, (m+1)\Delta, \Delta, (m+2)\Delta, 2\Delta, \dots, (m+j)\Delta, j\Delta, \dots, (m+m)\Delta, m\Delta\} \% 2^{-i-1} \\ &= \{0, 2(m+1)\delta, 2\delta, 2(m+2)\delta, 4\delta, \dots, 2(m+j)\delta, 2j\delta, \dots, 2(m+m)\delta, 2m\delta\} \% 2^{-i-1} \\ &= \{0, \delta, 2\delta, 3\delta, 4\delta, \dots, (2j-1)\delta, 2j\delta, \dots, (2m-1)\delta, 2m\delta\}, \end{aligned}$$

since  $2(m+j)\delta \% 2^{-i-1} = (2m+1+2j-1)\delta \% 2^{-i-1} = (2j-1)\delta$ , for  $j = 1, \dots, m$ . ◀

► **Lemma 7 (Restated).** *Consider any two points  $p, q \in [0, 1]^d$ , and let  $\mathcal{T}$  be the infinite quadtree of  $[0, 2]^d$ . For  $D = 2 \lceil d/2 \rceil$  and  $i = 0, \dots, D$ , let  $v_i = (i/(D+1), \dots, i/(D+1))$ . Then there exists an  $i \in \{0, \dots, D\}$ , such that  $p + v_i$  and  $q + v_i$  are contained in a cell of  $\mathcal{T}$  with side length  $\leq 2(D+1) \|p - q\|$ .*

**Proof.** Without loss of generality, suppose  $d$  is even (as such  $D = d$ ). Let  $\ell \in \mathbb{N}$ , such that for  $\alpha = 2^{-\ell}$ , we have

$$(d+1) \|p - q\| < \alpha \leq 2(d+1) \|p - q\|.$$

For  $\tau \in [0, 1]$ , let  $\mathbf{G} + \tau$  denote the (infinite) grid with sidelength  $\alpha$  shifted by the point  $(\tau, \dots, \tau)$ .

Let  $X = \{i/(d+1) \mid i = 0, \dots, d\}$  be the set of shifts considered. Since we are shifting a grid with sidelength  $\alpha$ , the shifting is periodical with value  $\alpha$ . It is thus sufficient to consider the shifts modulo  $\alpha$ .

Let  $p = (p_1, \dots, p_d)$  and  $q = (q_1, \dots, q_d)$ . Assume that  $p_1 \leq q_1$ . A shift  $\tau$  is bad, for the first coordinate, if there is an integer  $i$ , such that  $p_1 \leq \tau + i\alpha \leq q_1$ . The set of bad shifts in the interval  $[0, \alpha]$  is

$$B_1 = \{(p_1, q_1) + i\alpha \mid i \in \mathbb{Z}\} \cap [0, \alpha].$$

The set  $B_1$  is either an interval of length  $|p_1 - q_1| \leq \|p - q\| < \alpha/(d+1)$ , or two intervals (of the same total length) adjacent to 0 and  $\alpha$ . In either case,  $B_1$  can contain at most one point of  $\alpha X = X \% \alpha$ , since the distance between any two values of  $\alpha X$  is at least  $\alpha/(d+1)$ , by Lemma 18.

Namely, the first coordinate rules out at most one candidate shift in  $X \times \alpha$ . Repeating the above argument for all  $d$  coordinates, we conclude that there is at least one shift in  $\alpha X$  that is good for all coordinates. Let  $\beta = \alpha i / (d + 1) \in \alpha X$  this be good shift. Namely,  $p$  and  $q$  belongs the same cell of  $G + \beta$ . The final step is to observe that shifting the points by  $-\beta$ , instead of the grid by distance  $\beta$  has the same effect (and  $-\beta \times \alpha \in \alpha X$ ), and as such, the canonical cell containing both  $p$  and  $q$  is in the quadtree  $\mathcal{T}$  as desired, and the sidelength of this cell is  $\alpha$ . ◀



# Pseudorandom Generators from the Second Fourier Level and Applications to AC0 with Parity Gates

**Eshan Chattopadhyay**

Department of Computer Science, Cornell University, 107 Hoy Rd, Ithaca, NY, USA  
eshanc@cornell.edu

**Pooya Hatami**<sup>1</sup>

Department of Computer Science, University of Texas at Austin, 2317 Speedway, Austin, TX, USA  
pooyahat@gmail.com

**Shachar Lovett**<sup>2</sup>

Department of Computer Science, University of California San Diego, La Jolla, CA, USA  
slovett@ucsd.edu

**Avishay Tal**<sup>3</sup>

Department of Computer Science, Stanford University, 353 Serra Mall, Stanford, CA, USA  
avishay.tal@gmail.com

---

## Abstract

---

A recent work of Chattopadhyay et al. (CCC 2018) introduced a new framework for the design of pseudorandom generators for Boolean functions. It works under the assumption that the Fourier tails of the Boolean functions are uniformly bounded for all levels by an exponential function. In this work, we design an alternative pseudorandom generator that only requires bounds on the second level of the Fourier tails. It is based on a derandomization of the work of Raz and Tal (ECCC 2018) who used the above framework to obtain an oracle separation between BQP and PH.

As an application, we give a concrete conjecture for bounds on the second level of the Fourier tails for low degree polynomials over the finite field  $\mathbb{F}_2$ . If true, it would imply an efficient pseudorandom generator for  $AC^0[\oplus]$ , a well-known open problem in complexity theory. As a stepping stone towards resolving this conjecture, we prove such bounds for the first level of the Fourier tails.

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

**Keywords and phrases** Derandomization, Pseudorandom generator, Explicit construction, Random walk, Small-depth circuits with parity gates

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2019.22

**Acknowledgements** We would like to thank Gil Cohen, Russell Impagliazzo, Valentine Kabanets, James Lee, Ran Raz, Rahul Santhanam, Roy Schwartz and Srikanth Srinivasan for very helpful conversations.

---

<sup>1</sup> Supported by a Simons Investigator Award #409864, David Zuckerman.

<sup>2</sup> Supported by NSF grant CCF-1614023.

<sup>3</sup> Supported by a Motwani Postdoctoral Fellowship and by NSF grant CCF-1763299.



## 1 Introduction

Pseudorandom generators are widely studied in computational complexity theory. The main focus of this paper is a new framework for the design of pseudorandom generators (abbrv. PRGs) based on Fourier tails, introduced recently by Chattopadhyay et al. [2]. We refer to the survey of Vadhan [14] for an introduction to pseudorandomness in complexity theory, and assume basic knowledge with common concepts.

Let  $\mathcal{F}$  be a family of  $n$ -variate Boolean functions, which is closed under restrictions. Namely, for any  $f \in \mathcal{F}$ , if we restrict some of the inputs of  $f$  to Boolean values, then the restricted function is also in  $\mathcal{F}$ . Nearly all classes of Boolean functions studied in the literature satisfy this property.

Given an  $n$ -variate Boolean function  $f$ , its level- $k$  Fourier tails for  $k = 1, \dots, n$  are defined as

$$\mathcal{L}_{1,k}(f) = \sum_{S \subseteq [n]: |S|=k} |\widehat{f}(S)|.$$

For a function class  $\mathcal{F}$  of  $n$ -variate Boolean functions define

$$\mathcal{L}_{1,k}(\mathcal{F}) = \max_{f \in \mathcal{F}} \mathcal{L}_{1,k}(f).$$

Chattopadhyay et al. [2] proved a general theorem, which constructs an explicit PRG for functions in  $\mathcal{F}$ , assuming that  $\mathcal{F}$  has bounded  $k$ -level Fourier tails for all  $k$ . This property is known to hold for many classes of interest (read-once branching programs of bounded width, low-depth circuits, low sensitivity functions, and more; see [2] for details).

► **Theorem 1** ([2]). *Let  $\mathcal{F}$  be a family of  $n$ -variate Boolean functions that is closed under restrictions. Assume that for some  $a, b \geq 1$  it holds that*

$$\mathcal{L}_{1,k}(\mathcal{F}) \leq a \cdot b^k \quad \forall k = 1, \dots, n.$$

*Then for any  $\varepsilon > 0$ , there exists an explicit PRG for  $\mathcal{F}$  with error  $\varepsilon$  and seed length  $s = b^2 \cdot \text{polylog}(an/\varepsilon)$ .*

Note that for any  $n$ -variate Boolean function one can take  $a = 1, b = \sqrt{n}$ , and hence the quadratic dependence on  $b$  in the seed length is optimal.

The main objective of this current work is to investigate whether PRGs can also be obtained from weaker assumptions on the Fourier tail. Specifically, whether it suffices that  $\mathcal{L}_{1,k}(\mathcal{F})$  is bounded for a few values of  $k$ , instead of for the full regime of  $k = 1, \dots, n$  as was required by [2]. Our main result is that this is indeed the case: it suffices to obtain bounds for the second level of the Fourier tail.

► **Theorem 2** (Main result, informal version). *Let  $\mathcal{F}$  be a family of  $n$ -variate Boolean functions closed under restrictions. Assume that for some  $t \geq 1$  it holds that*

$$\mathcal{L}_{1,2}(\mathcal{F}) \leq t.$$

*Then for any  $\varepsilon > 0$ , there exists an explicit PRG for  $\mathcal{F}$  with error  $\varepsilon$  and seed length  $\text{poly}(t, \log n, 1/\varepsilon)$ .*

For a more precise formula for the seed length see Theorem 6. We note that the dependence on the error parameter  $\varepsilon$  in Theorem 2 is much worse compared to Theorem 1 – polynomial instead of polylogarithmic. We discuss this in Section 4.



## 1.1 A potential PRG for $\mathbb{F}_2$ -polynomials and $\text{AC}^0[\oplus]$

There are known deep relationships between the ability to construct explicit pseudorandom generators, and the ability to prove correlation bounds, for many classes of Boolean functions. One of the few classes where the latter is known but the former is not is  $\text{AC}^0[\oplus]$ , which is the classes of constant-depth polynomial-size Boolean circuits with AND, OR, NOT and PARITY gates. Classical works of Razborov [10] and Smolensky [11] prove that this class cannot approximate the MAJORITY function. On the other hand, the problem of constructing explicit PRGs for  $\text{AC}^0[\oplus]$  is a well-known open problem in complexity theory. We refer to the survey of Viola [15] for further discussion on this challenge.

We give a concrete (and plausible in our minds) conjecture which, combined with Theorem 2, would imply such a PRG. Let  $\text{Poly}_{n,d}$  denote the class of  $n$ -variate Boolean functions which are computed by  $\mathbb{F}_2$ -polynomials of degree at most  $d$ . This class is clearly closed under restrictions.

► **Conjecture 3.**  $\mathcal{L}_{1,2}(\text{Poly}_{n,d}) = O(d^2)$ . That is, if  $p : \mathbb{F}_2^n \rightarrow \mathbb{F}_2$  is a polynomial of degree  $d$ , and  $f(x) = (-1)^{p(x)}$ , then

$$\sum_{i,j \in [n], i < j} |\widehat{f}(\{i,j\})| = O(d^2).$$

A corollary of Conjecture 3, when combined with Theorem 2, is the construction of explicit PRGs for degree- $d$  polynomials over  $\mathbb{F}_2$  with seed length  $\text{poly}(\log n, d, 1/\varepsilon)$ . This would be a major breakthrough in complexity theory, as currently no PRGs are known for polynomials of degree  $d = \Omega(\log n)$ . We note that a similar seed length would follow from a weaker version of Conjecture 3 with the bound  $\mathcal{L}_{1,2}(\text{Poly}_{n,d}) \leq \text{poly}(\log n, d)$ . However, we conjecture that  $O(d^2)$  is the correct bound.

We further note that such PRGs would directly imply PRGs for  $\text{AC}^0[\oplus]$ .

► **Claim 4.** Assume that Conjecture 3 holds. Then, for any  $\varepsilon > 0$  there exists an explicit PRG for the class of  $\text{AC}^0[\oplus]$ -circuits of size  $s$  and depth  $e$  on  $n$  inputs, with error  $\varepsilon$  and seed length  $\text{poly}((\log(s/\varepsilon))^e, \log n, 1/\varepsilon)$ .

In particular, assuming the conjecture, for any constants  $e$  and  $c$ ,  $\text{AC}^0[\oplus]$ -circuits of size at most  $n^c$  and depth at most  $e$  have a PRG with seed length  $\text{poly}(\log n, 1/\varepsilon)$  and error  $\varepsilon$ .

**Proof sketch.** Let  $f : \{0,1\}^n \rightarrow \{0,1\}$  be computed by an  $\text{AC}^0[\oplus]$  circuit of size  $s$  and depth  $e$ . Razborov [10] and Smolensky [12] proved that there exists a distribution over polynomials  $p : \mathbb{F}_2^n \rightarrow \mathbb{F}_2$  of degree  $d = O(\log(s/\varepsilon))^e$  such that for each  $x \in \{0,1\}^n$ ,  $\Pr_p[p(x) \neq f(x)] \leq \varepsilon/3$ . Theorem 2 gives a PRG for polynomials of degree  $d$  with error  $\varepsilon/3$  and seed length  $\text{poly}(\log n, d, 1/\varepsilon)$ . By the Razborov-Smolensky result, this PRG is also a PRG for  $f$  with error  $\varepsilon$ . ◀

### Evidence Supporting The Conjecture

We present three results that supports the validity of Conjecture 3:

1. As a stepping stone towards resolving Conjecture 3, we prove a bound on the first level of the Fourier tail of low degree polynomials over  $\mathbb{F}_2$ .

► **Theorem 5.**  $\mathcal{L}_{1,1}(\text{Poly}_{n,d}) \leq 4d$ .

2. We note that as a special case of the main result in [3], any read-once  $\mathbb{F}_2$ -polynomial  $f$  (i.e., a sum of monomials on disjoint sets of variables) has

$$\mathcal{L}_{1,2}(f) \leq O(\log n)^8.$$

Getting a similar poly  $\log(m)$  bound for **general** polynomials with  $m$  monomials, would be sufficient for the application in Claim 4.

3. In [2] the following exponential bound was proved on the  $\mathcal{L}_{1,2}(\text{Poly}_{n,d})$ :

$$\mathcal{L}_{1,2}(\text{Poly}_{n,d}) \leq 4 \cdot 2^{6d}.$$

(as a special case of a bound on  $\mathcal{L}_{1,k}$  more generally.) We remark that this bound depends only on  $d$  and not on  $n$ . Thus, if our conjecture is false, any counter-example to it must have degree  $d = \omega(1)$ .

### Organization

We prove Theorem 2 in Section 2. We prove Theorem 5 in Section 3. We discuss further research in Section 4.

## 2 PRG from level two Fourier bounds

The main result of this section is an explicit pseudorandom generator for Boolean functions that have bounded Fourier tails on the second level.

► **Theorem 6.** *Let  $\mathcal{F}$  be a family of  $n$ -variate Boolean functions closed under restrictions. Assume that for some  $t \geq 1$  it holds that*

$$\mathcal{L}_{1,2}(\mathcal{F}) \leq t.$$

*Then for any  $\varepsilon > 0$ , there exists an explicit PRG for  $\mathcal{F}$  with error  $\varepsilon$  and seed length  $O((t/\varepsilon)^{2+o(1)} \cdot \text{polylog}(n))$ .*

The framework to construct the PRG is similar to the one used in [2]. The first step is to construct a fractional PRG for  $\mathcal{F}$  that is  $p$ -noticeable. Now using the *polarizing* random walk technique used in [2], we convert this fractional PRG into the required standard PRG. Our fractional PRG is based on ideas developed in [9]. We first recall the basic framework of [2].

### Pseudorandom generators

Let  $f : \{-1, 1\}^n \rightarrow \{-1, 1\}$  be a Boolean function. A PRG for  $f$  with error  $\varepsilon$  is a random variable  $X \in \{-1, 1\}^n$  such that

$$|\mathbf{E}[f(X)] - \mathbf{E}[f(U_n)]| \leq \varepsilon,$$

where  $U_n$  is the uniform distribution in  $\{-1, 1\}^n$ . It has seed length  $s$  if  $X$  can be sampled as

$$X = G(U_s)$$

where  $G : \{-1, 1\}^s \rightarrow \{-1, 1\}^n$  is an explicit function<sup>4</sup>.

$X$  is a PRG for a class of functions  $\mathcal{F}$  if it is a PRG for each function  $f \in \mathcal{F}$ .

---

<sup>4</sup> There are various notions of explicitness used in the complexity literature. For our purposes any notion would do.

### Fractional pseudorandom generators

Let  $f : \{-1, 1\}^n \rightarrow \{-1, 1\}$  be a Boolean function. It has a unique multi-linear extension as  $f : [-1, 1]^n \rightarrow [-1, 1]$ . A fractional PRG for  $f$  with error  $\varepsilon$  is a random variable  $X \in [-1, 1]^n$  such that

$$|\mathbf{E}[f(X)] - f(\vec{0})| \leq \varepsilon.$$

Note that  $f(\vec{0}) = \mathbf{E}[f(U_n)]$ . It has seed length  $s$  if  $X$  can be sampled as

$$X = G(U_s)$$

where  $G : \{-1, 1\}^s \rightarrow [-1, 1]^n$  is an explicit function. The fractional PRG  $X$  is  $p$ -noticeable if

$$\mathbf{E}[X_i^2] \geq p \quad \forall i = 1, \dots, n.$$

### From fractional PRGs to PRGs

The following is the main result of [2], which converts a fractional PRG into a standard PRG.

► **Theorem 7** ([2]). *Let  $\mathcal{F}$  be a family of  $n$ -variate Boolean functions that is closed under restrictions. Let  $X$  be a  $p$ -noticeable fractional pseudorandom generator for  $\mathcal{F}$  with seed length  $s$  and error  $\varepsilon$ . Then, there exists a pseudorandom generator for  $\mathcal{F}$  with seed length  $O(s \cdot \log(n/\varepsilon)/p)$  and error  $O(\varepsilon \cdot \log(n/\varepsilon)/p)$ .*

Given, the above theorem, the missing piece to get the desired PRG in Theorem 6 is to construct an appropriate fractional PRG. The following lemma achieves exactly this.

► **Lemma 8.** *Let  $\mathcal{F}$  be a family of  $n$ -variate Boolean functions closed under restrictions. Assume that for some  $t \geq 1$  it holds that*

$$\mathcal{L}_{1,2}(\mathcal{F}) \leq t.$$

*Then for any  $\varepsilon > 0$ , there exists an explicit  $p$ -noticeable fractional PRG for  $\mathcal{F}$  with error  $\varepsilon$  and seed length  $s$  where:*

$$\begin{aligned} 1/p &= O(\log(n/\varepsilon)) \\ s &= O((t/\varepsilon)^{2+o(1)} \cdot \log(n) \cdot \log(n/\varepsilon)). \end{aligned}$$

It is direct to obtain Theorem 6 from Theorem 7 and Lemma 8. We prove Lemma 8 in the remainder of this section.

As mentioned before, the fractional PRG is constructed based on ideas developed in [9]. In particular, our fractional PRG can be seen as a derandomization of the main distribution used in [9].

We first abstract and restate one of the main arguments in [9]. This abstraction appeared in a blog post of Boaz Barak and Jarosław Błasiok [1]. Below, we abbreviate a Multi-Variate Gaussian as MVG. Given a random variable  $Z \in \mathbb{R}^n$ , we denote by  $\text{trnc}(Z)$  its truncation to  $[-1, 1]^n$ . That is,  $\text{trnc}(Z)_i = \min(1, \max(-1, Z_i))$  for  $i \in [n]$ .

► **Theorem 9** ([9], restated). *Let  $n, t \geq 1, \delta \in (0, 1)$ . Let  $Z \in \mathbb{R}^n$  be a zero-mean MVG random variable with the following two properties:*

- (i) *For  $i \in [n]$ :  $\mathbf{Var}[Z_i] \leq \frac{1}{8 \ln(n/\delta)}$ .*
- (ii) *For  $i, j \in [n], i \neq j$ :  $|\mathbf{Cov}[Z_i, Z_j]| \leq \delta$ .*

*Let  $\mathcal{F}$  be a class of  $n$ -variate Boolean functions which is closed under restrictions. Assume that  $\mathcal{L}_{1,2}(\mathcal{F}) \leq t$ . Then for any  $f \in \mathcal{F}$  it holds that  $|\mathbf{E}[f(\text{trnc}(Z))] - f(\vec{0})| \leq O(\delta \cdot t)$ .*

## 22:6 Pseudorandom Generators from Second Level Fourier Bounds

For completeness, we prove Theorem 9 in the appendix – the proof basically repeats the argument in [9] but for a general multivariate Gaussian distribution, instead of the Forrelation distribution considered there. We now show how to use Theorem 9 to construct a  $p$ -noticeable fractional PRG for  $\mathcal{F}$  with error  $\varepsilon$  and seed length  $s$ , where  $1/p = O(\log(n/\varepsilon))$  and  $s = \text{poly}(t, \log(n), 1/\varepsilon)$ .

- I. We show that a MVG distribution with the parameters needed in Theorem 9 can be of rank  $\ell = \text{poly}(\log(n), t, 1/\varepsilon)$ . That is, we sample  $\ell$  independent  $\mathcal{N}(0, 1)$  random variables and apply an explicit linear transformation  $T : \mathbb{R}^\ell \rightarrow \mathbb{R}^n$  to get a random variable in  $\mathbb{R}^n$  that satisfies the two conditions of Theorem 9.
- II. We discretized the above process.

### Step I: Dimension Reduction

Let  $\delta = \varepsilon/t$  and let  $\ell$  be a parameter to be determined soon. Let  $\mathcal{C}$  be a code on  $\{0, 1\}^\ell$  with at least  $n$  codewords, such that  $\mathcal{C}$  is  $\delta$ -balanced. Namely, every codeword in  $\mathcal{C}$  has Hamming weight between  $(\frac{1}{2} - \delta)\ell$  and  $(\frac{1}{2} + \delta)\ell$ . Such a code can be obtained from explicit constructions of small-biased spaces over  $\{0, 1\}^\ell$ . The best known construction is by Ta-Shma [13] which achieves  $\ell = (\log n)/\delta^{2+o(1)}$ .

Set  $p = 1/(8 \ln(n/\delta))$ . Let  $c^1, \dots, c^n \in \mathcal{C}$  be distinct codewords. Define an  $n \times \ell$  matrix  $A$  given by

$$A_{i,j} = \sqrt{\frac{p}{\ell}} \cdot (-1)^{c_j^i},$$

where  $c^i = (c_1^i, \dots, c_\ell^i)$ . Let  $Y$  be a random vector in  $\mathbb{R}^\ell$  where each  $Y_i$  is an independent  $\mathcal{N}(0, 1)$  Gaussian. Define

$$Z = AY.$$

It is straightforward to verify from the construction that  $Z$  is a multivariate Gaussian distribution over  $\mathbb{R}^n$  with mean zero which satisfies that  $\mathbf{Var}[Z_i] = p$  for all  $i \in [n]$ , and  $|\mathbf{Cov}[Z_i, Z_j]| \leq \delta$  for all distinct  $i, j \in [n]$ .

### Step II: Discretizing the Randomness

We prove the following lemma, which allows to approximately sample a standard MVG  $Y \in \mathbb{R}^\ell$  as needed above using a few random bits.

► **Lemma 10.** *For any  $\ell, \eta > 0$  there exists  $s = O(\ell \cdot \log(\ell/\eta))$  and an explicit generator  $G : \{0, 1\}^s \rightarrow \mathbb{R}^\ell$  such that the following holds.*

*Let  $f : [-1, 1]^n \rightarrow [-1, 1]$  be a multi-linear function,  $A \in [-1, 1]^{n \times \ell}$  and  $Y$  be a random variable over  $\mathbb{R}^\ell$  where each  $Y_i$  is an independent  $\mathcal{N}(0, 1)$  Gaussian. Then*

$$|\mathbf{E}[f(\text{trnc}(AY))] - \mathbf{E}[f(\text{trnc}(AG(U_s)))]| \leq \eta(n + 2).$$

We say that a random variable  $W \in \mathbb{R}$  is a  $\lambda$ -approximate Gaussian if there is a correlated standard Gaussian  $W' \sim \mathcal{N}(0, 1)$  such that  $\mathbf{Pr}[|W - W'| > \lambda] < \lambda$ . We will use the following lemma of Kane [6] which shows how to approximate a Gaussian in a randomness efficient way.

► **Lemma 11 ([6]).** *There is an explicit construction of a  $\lambda$ -approximate Gaussian random variable using  $O(\log(1/\lambda))$  bits of randomness.*

The generator  $G$  would simply be  $\ell$  independent copies of a  $\lambda$ -approximate Gaussian given by the above lemma, with  $\lambda = \frac{\eta}{\ell}$ . We denote by  $Y' = G(U_s)$  and by  $Y$  the coupled standard MVG in  $\mathbb{R}^\ell$ .

Let  $\mathcal{E}$  denote the event that  $\|Y - Y'\|_\infty \leq \lambda$ . By a union bound,  $\Pr(\mathcal{E}) \geq 1 - \eta$ . Conditioned on  $\mathcal{E}$  it is easy to check that  $\|\text{trnc}(AY) - \text{trnc}(AY')\|_\infty \leq \eta$ . Finally, we use the multi-linearity and boundedness of  $f$  in the following lemma to finish the proof.

► **Lemma 12.** *Let  $f : [-1, 1]^n \rightarrow [-1, 1]$  be a multi-linear function. Then for every  $x, y \in [-1, 1]^n$  we have  $|f(x) - f(y)| \leq n \cdot \|x - y\|_\infty$ .*

**Proof.** For every  $i \in \{0, 1, \dots, n\}$  define  $z^{(i)} := (x_1, \dots, x_i, y_{i+1}, \dots, y_n)$ , note that  $z^{(n)} = x$  and  $z^{(0)} = y$ . We have

$$f(x) - f(y) = \sum_{i=1}^n f(z^{(i)}) - f(z^{(i-1)}).$$

Now note that since  $f$  is a multilinear function, for every  $i$ ,

$$|f(z^{(i)}) - f(z^{(i-1)})| = |h_i(x_i) - h_i(y_i)| \leq |x_i - y_i|,$$

where  $h_i(z) = f(x_1, \dots, x_{i-1}, z, y_{i+1}, \dots, y_n)$ . The above inequality holds as  $h_i$  is an affine function mapping  $[-1, 1]$  to  $[-1, 1]$ . We thus obtain that

$$|f(x) - f(y)| \leq \sum_{i=1}^n |x_i - y_i| \leq n \cdot \|x - y\|_\infty. \quad \blacktriangleleft$$

Using Lemma 12 and condition on the event  $\mathcal{E}$  we have

$$|f(\text{trnc}(AY)) - f(\text{trnc}(AY'))| \leq \eta n.$$

As  $f$  is bounded in  $[-1, 1]$  we obtain the bound

$$|\mathbf{E}[f(\text{trnc}(AY))] - \mathbf{E}[f(\text{trnc}(AY'))]| \leq \eta n + 2 \Pr[-\mathcal{E}] \leq \eta(n + 2).$$

### Completing the proof

We put things together to finish the proof of Lemma 8. Recall that  $\delta = \varepsilon/t$  and  $\ell = (\log n)/\delta^{2+o(1)} = (t/\varepsilon)^{2+o(1)} \cdot \log(n)$  from Step I. Set  $p = 1/(8 \ln(n/\delta)) = 1/(8 \ln(nt/\varepsilon))$ . Let  $A \in [-1, 1]^{n \times \ell}$  be the matrix constructed in step I. Set  $\eta = \varepsilon/(n + 2)$  and let  $G : \{0, 1\}^s \rightarrow \mathbb{R}^\ell$  be the generator constructed in step II. We take

$$X = AG(U_s).$$

The arguments above show that  $X$  is a fractional PRG for  $\mathcal{F}$  with error  $O(\varepsilon)$ . In addition,  $X$  is  $p$ -noticeable. To conclude we compute the seed length  $s$ :

$$s = O(\ell \cdot \log(\ell/\eta)) = O\left(\left(\frac{t}{\varepsilon}\right)^{2+o(1)} \cdot \log n \cdot \log(n/\varepsilon)\right).$$

### 3 Level one Fourier bounds for polynomials

In this section we bound the level one Fourier tail of low degree polynomials over  $\mathbb{F}_2$ .

► **Theorem 13.** *Let  $p : \mathbb{F}_2^n \rightarrow \mathbb{F}_2$  be a polynomial of degree  $d$ , and let  $f(x) = (-1)^{p(x)}$ . Then*

$$\mathcal{L}_{1,1}(f) = \sum_{i=1}^n |\widehat{f}(i)| \leq 4d.$$

**Proof.** We assume for simplicity that  $n$  is even; the proof is analogous for odd  $n$ . We have

$$\sum_{i=1}^n |\widehat{f}(i)| = \sum_{i=1}^n s_i \cdot \mathbf{E}_x [f(x)(-1)^{x_i}],$$

where  $s_i = \text{sign}(\widehat{f}(i))$ . We may assume without loss of generality that  $s_i = 1$  for all  $i$ , by replacing  $x_i$  with  $1 - x_i$  whenever  $s_i = -1$ . Thus, it suffices to upper bound

$$E := \mathbf{E}_x \left[ f(x) \sum_{i=1}^n (-1)^{x_i} \right].$$

For  $t = 1, \dots, n/2$  define the functions  $T_t : \{0, 1\}^n \rightarrow \{-1, 0, 1\}$  as follows:

$$T_t(x) := \begin{cases} -1 & \text{if } \sum x_i \geq n/2 + t \\ 1 & \text{if } \sum x_i \leq n/2 - t \\ 0 & \text{otherwise} \end{cases}.$$

Then

$$E = 2 \sum_{t=1}^{n/2} \mathbf{E}_x [f(x)T_t(x)].$$

We need a few more definitions. Let  $U_t := \{x \in \{0, 1\}^n : |\sum x_i - n/2| \geq t\}$ . Define  $M_t : U_t \rightarrow \mathbb{F}_2$  as  $M_t(x) = 0$  if  $\sum x_i \geq n/2 + t$ , and  $M_t(x) = 1$  if  $\sum x_i \leq n/2 - t$ . Note that  $T_t(x) = (-1)^{M_t(x)}$  for  $x \in U_t$ , and  $T_t(x) = 0$  for  $x \notin U_t$ . Let  $A_t := \{x \in U_t : p(x) = M_t(x)\}$ . Then

$$e_t := \mathbf{E}_x [f(x)T_t(x)] = \frac{2|A_t| - |U_t|}{2^n}.$$

We next apply a dimension argument similar to that used by Razborov [10] and Smolensky [11] (we adopt a Kopparty's presentation of the argument [7, Lemma 6]). Consider the space of functions  $g : A_t \rightarrow \mathbb{F}_2$ . On the one hand, its dimension is  $|A_t|$ . On the other hand, any function  $g : U_t \rightarrow \mathbb{F}_2$  can be decomposed as

$$g(x) = g_1(x)M_t(x) + g_2(x),$$

where  $g_1, g_2$  are polynomials over  $\mathbb{F}_2$  of degree  $\leq n/2 - t$ . Thus, any function  $g : A_t \rightarrow \mathbb{F}_2$  can be expressed as a polynomial  $g(x) = g_1(x)p(x) + g_2(x)$  which is of degree  $\leq n/2 - t + d$ . Thus, we can bound  $|A_t|$  by the dimension of this linear space of polynomials,

$$|A_t| \leq \sum_{i=0}^{n/2-t+d} \binom{n}{i}.$$

Using the fact that  $|U_t| = 2 \sum_{i=0}^{n/2-t} \binom{n}{i}$  we can upper bound  $e_t$  by

$$e_t \leq \frac{2 \sum_{i=1}^d \binom{n}{n/2-t+i}}{2^n}.$$

We thus can bound  $E$  by

$$E = 2 \sum_{t=1}^{n/2} e_t \leq 4 \sum_{t=1}^{n/2} \sum_{i=1}^d \frac{\binom{n}{n/2-t+i}}{2^n} = 4 \sum_{i=1}^d \sum_{j=0}^{n/2-1} \frac{\binom{n}{j+i}}{2^n} \leq 4d. \quad \blacktriangleleft$$

### 3.1 Level two Fourier bounds from level one bounds

We present a simple argument showing that for any family  $\mathcal{F}$  of  $n$ -variate Boolean functions that is closed under restrictions, a bound of  $\mathcal{L}_{1,1}(\mathcal{F}) \leq t$  implies  $\mathcal{L}_{1,2}(\mathcal{F}) \leq O(t \cdot \sqrt{n \log n})$ . Using this connection, we get that polynomials of degree  $\text{polylog}(n)$  have  $\mathcal{L}_{1,2}(\cdot)$  at most  $\sqrt{n} \cdot \text{polylog}(n)$ . Recall that we conjecture that the right bound should be  $\text{polylog}(n)$  (i.e., exponentially smaller). Nevertheless, even improving this bound slightly to  $n^{1/2-o(1)}$  would imply a non-trivial PRG fooling  $\text{polylog}(n)$ -degree  $\mathbb{F}_2$ -polynomials and  $\text{AC}^0[\oplus]$  circuits with seed-length  $n^{1-o(1)}$ . In comparison, the current state of the art PRG for  $\text{AC}^0[\oplus]$  circuits has seed-length  $n - n/\text{polylog}(n)$  [4].

► **Claim 14.** *Let  $\mathcal{F}$  be a class of  $n$ -variate Boolean functions that is closed under restrictions. Let  $t \geq 1$ . Assume that  $\mathcal{L}_{1,1}(\mathcal{F}) \leq t$ . Then,  $\mathcal{L}_{1,2}(\mathcal{F}) \leq t \cdot O(\sqrt{n \log n})$ .*

**Proof.** Let  $f : \{-1, 1\}^n \rightarrow \{-1, 1\}$  be some Boolean function in  $\mathcal{F}$ . We bound  $\mathcal{L}_{1,2}(f) = \sum_{i < j} |\hat{f}(i, j)|$ . We begin by partitioning the set of coordinates of  $f$  into two disjoint parts  $[n] = X \cup Y$  and summing only the cross-terms  $L_1(X, Y) = \sum_{i \in X} \sum_{j \in Y} |\hat{f}(i, j)|$ . We note that there exists a partition  $[n] = X \cup Y$  such that  $L_1(X, Y) \geq \mathcal{L}_{1,2}(f)/2$ . This holds since a random partition has on expectation

$$\mathbf{E}_{X, Y} [L_1(X, Y)] = \sum_{i < j} |\hat{f}(i, j)| \cdot (\Pr[i \in X, j \in Y] + \Pr[i \in Y, j \in X]) = \mathcal{L}_{1,2}(f) \cdot \frac{1}{2}.$$

Fix a partition  $(X, Y)$  for which  $L_1(X, Y) \geq \mathcal{L}_{1,2}(f) \cdot \frac{1}{2}$ . In the remainder, we bound

$$L_1(X, Y) = \sum_{i \in X, j \in Y} |\hat{f}(\{i, j\})| = \sum_{i \in X, j \in Y} s_{i, j} \cdot \hat{f}(\{i, j\})$$

for some sign matrix  $s \in \{-1, 1\}^{X \times Y}$ . For any fixed  $x \in \{-1, 1\}^X$  we denote by  $f_x : \{-1, 1\}^Y \rightarrow \{-1, 1\}$  the function defined by  $f_x(y) = f(x, y)$ . Note that  $f_x$  is a restriction of  $f$  thus by our assumption, its  $\mathcal{L}_{1,1}$  is at most  $t$ . We get

$$\begin{aligned} L_1(X, Y) &= \mathbf{E}_{\substack{x \in \{-1, 1\}^X, \\ y \in \{-1, 1\}^Y}} \left[ \sum_{i \in X, j \in Y} s_{i, j} \cdot f(x, y) \cdot x_i \cdot y_j \right] \\ &= \mathbf{E}_{x \in \{-1, 1\}^X} \left[ \sum_{i \in X, j \in Y} s_{i, j} \cdot x_i \cdot \mathbf{E}_{y \in \{-1, 1\}^Y} [f(x, y) \cdot y_j] \right] \\ &= \mathbf{E}_{x \in \{-1, 1\}^X} \left[ \sum_{i \in X, j \in Y} s_{i, j} \cdot x_i \cdot \hat{f}_x(\{j\}) \right] = \mathbf{E}_{x \in \{-1, 1\}^X} \left[ \sum_{j \in Y} \left( \hat{f}_x(\{j\}) \cdot \sum_{i \in X} s_{i, j} \cdot x_i \right) \right] \\ &\leq \mathbf{E}_{x \in \{-1, 1\}^X} \left[ \sum_{j \in Y} |\hat{f}_x(\{j\})| \cdot \left| \sum_{i \in X} s_{i, j} \cdot x_i \right| \right] \leq \mathbf{E}_{x \in \{-1, 1\}^X} \left[ t \cdot \max_{j \in Y} \left| \sum_{i \in X} s_{i, j} \cdot x_i \right| \right] \end{aligned}$$

By Chernoff's bounds, the expectation of  $\max_{j \in Y} |\sum_{i \in X} s_{i,j} \cdot x_i|$  is at most  $O(\sqrt{n \log n})$ . Thus overall  $\mathcal{L}_{1,2}(f) \leq 2 \cdot L_1(X, Y) \leq 2t \cdot O(\sqrt{n \cdot \log n})$ . ◀

#### 4 Further research

A clear advantage of Theorem 2 over Theorem 1 is that only bounds on the second level of the Fourier tails are needed, instead of bounds for all levels. However, we pay a price, as the dependence on the error parameter  $\varepsilon$  is polynomial instead of poly-logarithmic. This raises a natural problem: can a better dependency on  $\varepsilon$  be obtained if the Fourier tails are assumed to be bounded for several levels  $k$ ? In particular, information on how many levels is needed in order to obtain poly-logarithmic dependency on the error  $\varepsilon$ ? We leave these questions to future work.

---

#### References

- 1 Boaz Barack and Jarosław Błasiok. On the Raz-Tal oracle separation of BQP and PH, 2018. <https://windowsontheory.org/2018/06/17/on-the-raz-tal-oracle-separation-of-bqp-and-ph/>.
- 2 Eshan Chattopadhyay, Pooya Hatami, Kaave Hosseini, and Shachar Lovett. Pseudorandom Generators from Polarizing Random Walks. In *33rd Computational Complexity Conference, CCC 2018*, pages 1:1–1:21, 2018.
- 3 Eshan Chattopadhyay, Pooya Hatami, Omer Reingold, and Avishay Tal. Improved pseudorandomness for unordered branching programs through local monotonicity. In *STOC*, pages 363–375. ACM, 2018.
- 4 Bill Fefferman, Ronen Shaltiel, Christopher Umans, and Emanuele Viola. On Beating the Hybrid Argument. *Theory of Computing*, 9:809–843, 2013.
- 5 Leon Isserlis. On a formula for the product-moment coefficient of any order of a normal frequency distribution in any number of variables. *Biometrika*, 12(1/2):134–139, 1918.
- 6 Daniel M. Kane. A Polylogarithmic PRG for Degree 2 Threshold Functions in the Gaussian Setting. In *Conference on Computational Complexity*, volume 33 of *LIPICs*, pages 567–581, 2015.
- 7 Swastik Kopparty. On the complexity of powering in finite fields. In *STOC*, pages 489–498. ACM, 2011.
- 8 Peter Mörters and Yuval Peres. *Brownian motion*, volume 30. Cambridge University Press, 2010.
- 9 Ran Raz and Avishay Tal. Oracle Separation of BQP and PH. *Electronic Colloquium on Computational Complexity (ECCC)*, 25:107, 2018.
- 10 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.
- 11 R. Smolensky. On representations by low-degree polynomials. In *FOCS*, pages 130–138, 1993.
- 12 Roman Smolensky. Algebraic Methods in the Theory of Lower Bounds for Boolean Circuit Complexity. In *STOC*, pages 77–82. ACM, 1987.
- 13 Amnon Ta-Shma. Explicit, almost optimal, epsilon-balanced codes. In *STOC*, pages 238–251, 2017.
- 14 Salil P. Vadhan. Pseudorandomness. *Foundations and Trends in Theoretical Computer Science*, 7(1-3):1–336, 2012.
- 15 Emanuele Viola. Guest Column: correlation bounds for polynomials over  $\{0, 1\}$ . *ACM SIGACT News*, 40(1):27–44, 2009.



## A Proof of Theorem 9

Throughout this section we take  $G$  to be a multivariate Gaussian distribution with zero mean, covariances at most  $\delta$  and variances at most 1. That is, if  $G = (G_1, \dots, G_n)$  then  $\mathbf{E}[G_i] = 0$ ,  $\mathbf{E}[G_i^2] \leq 1$  and  $|\mathbf{E}[G_i G_j]| \leq \delta$  for  $i \neq j$ .

### A.1 Preliminaries

Let  $f : \mathbb{R}^n \rightarrow \mathbb{R}$  be a multi-linear function, defined by

$$f(z) = \sum_{S \subseteq [n]} \widehat{f}(S) \cdot \prod_{i \in S} z_i, \quad (1)$$

where  $\widehat{f}(S) \in \mathbb{R}$ . We bound the difference between  $\mathbf{E}_{z \sim G}[f(\text{trnc}(pz))]$  and  $\mathbf{E}_{z \sim G}[f(pz)]$  for a small  $p \in (0, 1)$ . Note that whenever  $z' \in [-1, 1]^n$ , there is no difference between  $f(z')$  and  $f(\text{trnc}(z'))$ , and we only need to bound the difference when  $z'$  is outside  $[-1, 1]^n$ . The next claim bounds the value of  $|f(z')|$  when  $z'$  is outside  $[-1, 1]^n$ .

► **Claim 15** ([9, Claim 5.1]). *Let  $f : \mathbb{R}^n \rightarrow \mathbb{R}$  be a multi-linear function that maps  $\{-1, 1\}^n$  to  $[-1, 1]$ . Let  $z' \in \mathbb{R}^n$ . Then,  $|f(z')| \leq \prod_{i=1}^n \max(1, |z'_i|)$ .*

For  $\alpha \in (0, 1)$ ,  $z \in \mathbb{R}^n$ , we get that the value of  $|f(\alpha z)|$  is bounded by  $\prod_i \max(1, |\alpha z_i|)$ . The following claim bounds the latter times the indicator that  $\alpha z \neq \text{trnc}(\alpha z)$ .

► **Claim 16.** *Let  $\alpha \in (0, 1/\sqrt{4n}]$ . Then,*

$$\mathbf{E}_{z \sim G} \left[ \prod_{i=1}^n \max(1, |\alpha z_i|) \cdot \mathbb{1}_{\alpha z \neq \text{trnc}(\alpha z)} \right] \leq \sum_{k=1}^{\infty} e^{-k/(4\alpha^2 n)} \cdot n^k.$$

**Proof.** For  $i \in [n]$  and  $a_i \in \mathbb{N}$ , we consider the event

$$a_i \leq |\alpha \cdot z_i| < a_i + 1,$$

denoted by  $\mathcal{E}_{i, a_i}$ . Since each  $z_i$  is a Gaussian with mean 0 and variance at most 1, we have  $\Pr[\mathcal{E}_{i, a_i}] \leq e^{-a_i^2/(2\alpha^2)}$ . Using Claim 15 we have

$$\begin{aligned} (*) &= \mathbf{E}_{z \sim G} \left[ \prod_{i=1}^n \max(1, |\alpha z_i|) \cdot \mathbb{1}_{\alpha z \neq \text{trnc}(\alpha z)} \right] \\ &\leq \sum_{\vec{a} \in \mathbb{N}^n, \vec{a} \neq 0^n} \Pr[\wedge_{i=1}^n \mathcal{E}_{i, a_i}] \cdot \prod_{i=1}^n (1 + a_i) \\ &\leq \sum_{\vec{a} \in \mathbb{N}^n, \vec{a} \neq 0^n} \min_{i \in [n]} \{\Pr[\mathcal{E}_{i, a_i}]\} \cdot \prod_{i=1}^n (1 + a_i) \\ &\leq \sum_{\vec{a} \in \mathbb{N}^n, \vec{a} \neq 0^n} \prod_{i=1}^n \Pr[\mathcal{E}_{i, a_i}]^{1/n} \cdot \prod_{i=1}^n (1 + a_i) \end{aligned} \quad (2)$$

We bound

$$\Pr[\mathcal{E}_{i, a_i}]^{1/n} \cdot (1 + a_i) \leq e^{-a_i^2/(2\alpha^2 n)} \cdot (1 + a_i) \leq e^{-a_i^2/(4\alpha^2 n)}$$

22:12 Pseudorandom Generators from Second Level Fourier Bounds

since  $1 + a_i \leq e^{a_i} \leq e^{a_i^2/(4\alpha^2 n)}$  for  $\alpha^2 \leq 1/4n$ . We plug this estimate in Equation (2):

$$\begin{aligned}
 (*) &\leq \sum_{\vec{a} \in \mathbb{N}^n, \vec{a} \neq 0^n} e^{-\sum_i a_i^2/(4\alpha^2 n)} \\
 &\leq \sum_{k=1}^{\infty} e^{-k/(4\alpha^2 n)} \cdot \left| \left\{ \vec{a} \in \mathbb{N}^n : \sum_i a_i = k \right\} \right| \\
 &= \sum_{k=1}^{\infty} e^{-k/(4\alpha^2 n)} \cdot \binom{n+k-1}{k} \leq \sum_{k=1}^{\infty} e^{-k/(4\alpha^2 n)} \cdot n^k. \quad \blacktriangleleft
 \end{aligned}$$

► **Claim 17.** Let  $p \leq 1/4\sqrt{n}$ . Let  $f : \mathbb{R}^n \rightarrow \mathbb{R}$  be a multi-linear function that maps  $\{-1, 1\}^n$  to  $[-1, 1]$ . Let  $v \in [-1/2, 1/2]^n$ . Then,

$$\mathbf{E}_{z \sim G} [|f(\text{trnc}(v + p \cdot z)) - f(v + p \cdot z)|] \leq 2 \cdot \sum_{k=1}^{\infty} e^{-k/(16p^2 n)} \cdot n^k.$$

**Proof.** Let  $\mathcal{E}$  be the event that  $\text{trnc}(v + p \cdot z) \neq v + p \cdot z$ . Note that  $\mathcal{E}$  implies the event  $2pz \neq \text{trnc}(2pz)$  since  $v \in [-1/2, 1/2]^n$ . Using Claim 15, we get

$$\begin{aligned}
 \mathbf{E}_{z \sim G} [|f(\text{trnc}(v + p \cdot z)) - f(v + p \cdot z)|] &\leq \mathbf{E}_{z \sim G} [(1 + |f(v + p \cdot z)|) \cdot \mathbb{1}_{\mathcal{E}}] \\
 &\leq \mathbf{E}_{z \sim G} [(1 + |f(v + p \cdot z)|) \cdot \mathbb{1}_{2pz \neq \text{trnc}(2pz)}] \\
 &\leq \mathbf{E}_{z \sim G} \left[ \left( 1 + \prod_{i=1}^n \max(1, |v_i + p \cdot z_i|) \right) \cdot \mathbb{1}_{2pz \neq \text{trnc}(2pz)} \right] \\
 &\leq \mathbf{E}_{z \sim G} \left[ 2 \cdot \prod_{i=1}^n \max(1, |v_i + p \cdot z_i|) \cdot \mathbb{1}_{2pz \neq \text{trnc}(2pz)} \right].
 \end{aligned}$$

However,  $\prod_{i=1}^n \max(1, |v_i + p \cdot z_i|) \leq \prod_{i=1}^n \max(1, 1/2 + p|z_i|) \leq \prod_{i=1}^n \max(1, 2p|z_i|)$ . Using Claim 16 with  $\alpha = 2p$ , we get

$$\begin{aligned}
 \mathbf{E}_{z \sim G} [|f(\text{trnc}(v + p \cdot z)) - f(v + p \cdot z)|] &\leq \mathbf{E}_{z \sim G} \left[ 2 \cdot \prod_{i=1}^n \max(1, 2p|z_i|) \cdot \mathbb{1}_{2pz \neq \text{trnc}(2pz)} \right] \\
 &\leq 2 \cdot \mathbf{E}_{z \sim G} \left[ \prod_{i=1}^n \max(1, \alpha|z_i|) \cdot \mathbb{1}_{\alpha z \neq \text{trnc}(\alpha z)} \right] \\
 &\leq 2 \cdot \sum_{k=1}^{\infty} e^{-k/(4\alpha^2 n)} \cdot n^k. \quad \blacktriangleleft
 \end{aligned}$$

► **Claim 18 (Application of Isserlis' Theorem).** Let  $G$  be a MVG distribution over  $\mathbb{R}^n$  with zero-mean and covariances at most  $\delta$ . For  $S \subseteq [n]$ , let  $\widehat{G}(S) = \mathbf{E}_{Z \sim G} [\prod_{i \in S} Z_i]$ . Then,

1.  $\widehat{G}(S) = 0$  if  $|S|$  is odd.
2.  $|\widehat{G}(S)| \leq (k-1)!! \cdot \delta^{k/2}$  if  $|S| = k$  is even.

**Proof.** Both items rely on Isserlis' Theorem [5] (See also [http://en.wikipedia.org/wiki/Isserlis'\\_theorem](http://en.wikipedia.org/wiki/Isserlis'_theorem)) that gives a formula for the moments of any zero-mean multi Gaussian distribution. Isserlis' Theorem [5] states that in a zero-mean multivariate Gaussian distribution  $Z_1, \dots, Z_n$ , for a sequence of indices  $(i_1, \dots, i_k) \in [n]$ , we have  $\mathbf{E}[Z_{i_1} \cdots Z_{i_k}] = 0$  if  $k$  is odd and  $\mathbf{E}[Z_{i_1} \cdots Z_{i_k}] = \sum \prod \mathbf{E}[Z_{i_r} Z_{i_\ell}]$ , where the notation  $\sum \prod$  means summing over all distinct ways of partitioning  $Z_{i_1}, \dots, Z_{i_k}$  into pairs and each summand is the product of the  $k/2$  pairs. If  $|S| = k$  is even, since the covariance of each pair in  $G$  is at most  $\delta$  in absolute value and there are at most  $(k-1)!!$  partitions to pairs, we get  $|\widehat{G}(S)| \leq (k-1)!! \cdot \delta^{k/2}$ . ◀

The next claim expresses the difference of a multi-linear function  $f$  on two vectors,  $v$  and  $v + z$ , as the expected difference of random restrictions of  $f$  on  $0$  and  $2z$ , provided that  $v \in [-1/2, 1/2]^n$ . Applying this lemma when the entries of  $z$  are infinitesimally small means that bounded variation of random restrictions of  $f$  around  $0$  implies bounded variation of  $f$  around any  $v \in [-1/2, 1/2]^n$ .

► **Claim 19** ([2, Claim 3.3], restated in [1]). *Let  $f$  be a multi-linear function on  $\mathbb{R}^n$  and  $v \in [-1/2, 1/2]^n$ . There exists a distribution over random restrictions  $\rho$  such that for any  $z \in \mathbb{R}^n$ ,*

$$f(v + z) - f(v) = \mathbf{E}_{\rho} [f_{\rho}(2 \cdot z) - f_{\rho}(\vec{0})].$$

**Proof.** Given  $v \in [-1/2, 1/2]^n$ , we define a distribution  $\mathcal{R}_v$  over restrictions  $\rho \in \{-1, 1, *\}^n$ , as follows. For each entry  $i \in [n]$  independently, we set  $\rho_i = 1$  with probability  $1/4 + v_i/2$ ,  $\rho_i = -1$  with probability  $1/4 - v_i/2$ , and  $\rho_i = *$  with probability  $1/2$ . Note that since  $v \in [-1/2, 1/2]^n$  all these probabilities are indeed non-negative.

Let  $\rho \sim \mathcal{R}_v$ . For any vector  $z \in \mathbb{R}^n$ , we define a vector  $\tilde{z} = \tilde{z}(z, \rho) \in \mathbb{R}^n$ , as follows:

$$\tilde{z}_i = \begin{cases} \rho_i & \text{if } \rho_i \in \{-1, 1\} \\ 2 \cdot z_i & \text{otherwise} \end{cases}$$

Thus, for a fixed  $z \in \mathbb{R}^n$ , the vector  $\tilde{z}$  is a random variable that depends on  $\rho$ . We show that for any fixed  $z \in \mathbb{R}^n$ , the distribution of the random variable  $\tilde{z}$  is a product distribution (over inputs in  $\mathbb{R}^n$ ), and the expectation of  $\tilde{z}$  is the vector  $v + z$ . Indeed, each coordinate  $\tilde{z}_i$  is independent of the other coordinates, and its expected value is

$$\mathbf{E}_{\rho \sim \mathcal{R}_v} [\tilde{z}_i] = v_i + z_i.$$

Hence, since  $f$  is multi-linear and  $\tilde{z}$  has a product distribution, by Equation (1),  $\mathbf{E}_{\rho \sim \mathcal{R}_v} [f(\tilde{z})] = f(v + z)$ . We get

$$f(v + z) - f(v) = \mathbf{E}_{\rho \sim \mathcal{R}_v} [f(\tilde{z}(z, \rho))] - \mathbf{E}_{\rho \sim \mathcal{R}_v} [f(\tilde{z}(\vec{0}, \rho))] = \mathbf{E}_{\rho \sim \mathcal{R}_v} [f(\tilde{z}(z, \rho)) - f(\tilde{z}(\vec{0}, \rho))]$$

However, for any fixed  $\rho$ , we have  $f(\tilde{z}(z, \rho)) = f_{\rho}(2z)$ , where  $f_{\rho}$  is attained from  $f$  by fixing the coordinates that were fixed in  $\rho$ , according to  $\rho$ . Thus,

$$f(v + z) - f(v) = \mathbf{E}_{\rho \sim \mathcal{R}_v} [f_{\rho}(2 \cdot z) - f_{\rho}(\vec{0})]. \quad \blacktriangleleft$$

## A.2 The Proof

► **Claim 20.** *Let  $f : \{-1, 1\}^n \rightarrow \{-1, 1\}$  be a Boolean function with  $\mathcal{L}_{1,2}(f) \leq t$ . Let  $p \leq 1/2n$ . Then,*

$$\left| \mathbf{E}_{Z \sim G} [f(pZ)] - f(\vec{0}) \right| \leq p^2 \cdot t \cdot \delta + O(p^4 \cdot n^4 \cdot \delta^2).$$

**Proof.** By Equation (1) and since  $f(\vec{0}) = \hat{f}(\emptyset)$ ,

$$\begin{aligned} \left| \mathbf{E}_{Z \sim G} [f(pZ)] - f(\vec{0}) \right| &= \left| \mathbf{E}_{Z \sim G} \left[ \sum_{\emptyset \neq S \subseteq [n]} \hat{f}(S) \cdot \prod_{i \in S} (p \cdot Z_i) \right] \right| \\ &= \left| \sum_{\emptyset \neq S \subseteq [n]} \hat{f}(S) \cdot p^{|S|} \cdot \mathbf{E}_{Z \sim G} \left[ \prod_{i \in S} Z_i \right] \right| \\ &\leq \sum_{k=1}^n p^k \cdot \left( \max_{S: |S|=k} |\hat{G}(S)| \right) \cdot \sum_{S \subseteq [n], |S|=k} |\hat{f}(S)| \end{aligned}$$

22:14 Pseudorandom Generators from Second Level Fourier Bounds

For odd  $k$ , Claim 18 gives  $\max_{S:|S|=k} |\widehat{G}(S)| = 0$ . For even  $k$ , we have  $\max_{S:|S|=k} |\widehat{G}(S)| \leq (k-1)!! \cdot \delta^{k/2}$  by Claim 18. Plugging these bounds in the above expression gives

$$\begin{aligned} \left| \mathbf{E}_{Z \sim G}[f(pZ)] - f(\vec{0}) \right| &\leq p^2 \cdot \delta \cdot \sum_{S:|S|=2} |\widehat{f}(S)| + \sum_{k \geq 4, k \text{ even}} p^k \cdot \delta^{k/2} \cdot (k-1)!! \cdot \sum_{S:|S|=k} |\widehat{f}(S)| \\ &\leq p^2 \cdot \delta \cdot t + \sum_{k \geq 4, k \text{ even}} p^k \cdot \delta^{k/2} \cdot (k-1)!! \cdot \binom{n}{k} \\ &\hspace{15em} (\mathcal{L}_{1,2}(f) \leq t \text{ and } \forall S: |\widehat{f}(S)| \leq 1) \\ &\leq p^2 \cdot \delta \cdot t + \sum_{k \geq 4, k \text{ even}} p^k \cdot \delta^{k/2} \cdot n^k \\ &\leq p^2 \cdot \delta \cdot t + O(p^4 \cdot n^4 \cdot \delta^2) \hspace{10em} (p \leq 1/2n) \end{aligned}$$

◀

► **Theorem 21** (Theorem 9, restated). *Let  $n \in \mathbb{N}$ ,  $\delta, \sigma \in (0, 1)$ . Let  $G$  be a zero-mean multivariate Gaussian distribution over  $\mathbb{R}^n$  where  $Z \sim G$  has the following two properties:*

1. For  $i \in [n]$ :  $\mathbf{Var}[Z_i] \leq \sigma^2$
2. For  $i, j \in [n], i \neq j$ :  $|\mathbf{Cov}[Z_i, Z_j]| \leq \delta$ .

*Let  $\mathcal{F}$  be a class of  $n$ -variate Boolean functions which is closed under restrictions. Assume that  $\mathcal{L}_{1,2}(\mathcal{F}) \leq t$ . Then for any  $f \in \mathcal{F}$  it holds that  $|\mathbf{E}[f(\text{trnc}(Z))] - f(\vec{0})| \leq 4\delta \cdot t + 4n \cdot e^{-1/8\sigma^2}$ .*

**Proof.** Let  $m \in \mathbb{N}$  be sufficiently large (in particular  $m \geq (4n)^4$ ) and  $p = 1/\sqrt{m}$ . Let  $Z^{(1)}, \dots, Z^{(m)} \sim G$ . We define  $m+1$  hybrids  $H_0, \dots, H_m$ . Let  $H_0 = \vec{0}$ . For  $i = 1, \dots, m$ , let  $H_i = p \cdot (Z^{(1)} + \dots + Z^{(i)})$ . We observe that  $H_m \sim G$ . This is true since  $H_m$  is a multivariate Gaussian with the same expectation and the same covariance matrix as  $Z \sim G$ . We can think of  $H_0, H_1, \dots, H_m$  as a  $n$ -dimensional random walk. We bound

$$|\mathbf{E}[f(\text{trnc}(H_m))] - f(\vec{0})|$$

by considering two cases depending on whether or not at some point in the random walk we stepped outside of  $[-1/2, 1/2]^n$ .

For  $i \in \{0, \dots, m\}$ , let  $\mathcal{E}_i$  be the event that  $H_i \in [-1/2, 1/2]^n$ . We show that  $\mathcal{E}_i$  happens with high probability. In fact, we show that  $\mathcal{E} = \mathcal{E}_1 \wedge \mathcal{E}_2 \wedge \dots \wedge \mathcal{E}_m$  happens with high probability, with no dependency on the number of steps  $m$ . The claim follows from known properties of Brownian motions. For  $j \in [n]$ , let  $\mathcal{D}^{(j)}$  be the event that there exists an  $i \in [m]$  with  $|(H_i)_j| > 1/2$ . Clearly  $\neg \mathcal{E} \equiv \mathcal{D}^{(1)} \vee \mathcal{D}^{(2)} \vee \dots \vee \mathcal{D}^{(n)}$ .

We show that for each  $j \in [n]$ ,  $\mathbf{Pr}[\mathcal{D}^{(j)}] \leq 4 \cdot e^{-8/\sigma^2}$  and then apply a union bound. Each  $\{(H_i)_j\}_{i=0}^m$  is a random walk with  $m$  steps, which can be viewed as a discretization of a one-dimensional Brownian motion. A standard one-dimensional Brownian motion (or Wiener process) is a random process  $\{B(t)\}_{t \geq 0}$  with the properties: (1)  $B(0) = 0$ , (2) for all  $t, s \geq 0$ ,  $B(t+s) - B(t)$  is independent of the past  $\{B(t')\}_{t' \leq t}$  (3) for all  $t, s \geq 0$ ,  $B(t+s) - B(t) \sim \mathcal{N}(0, s)$ . Let  $\sigma_j^2 := \mathbf{Var}[z_j]$ . We observe that if  $B$  is a standard one-dimensional Brownian motion, then  $\{B(\sigma_j^2 \cdot i/m)\}_{i=0}^m$  is distributed exactly as  $\{(H_i)_j\}_{i=0}^m$ . Let  $M(t) = \sup_{0 \leq s \leq t} B(s)$  and  $M'(t) = \inf_{0 \leq s \leq t} B(s)$ . It suffices to show that  $M(\sigma_j^2) \leq 1/2$  and  $M'(\sigma_j^2) \geq -1/2$  with high probability. Known results on Brownian motions state that  $\mathbf{Pr}[M'(t) < -1/2] = \mathbf{Pr}[M(t) > 1/2] = \mathbf{Pr}[|B(t)| > 1/2]$  (cf. [8, Theorem 2.21]). The latter is at most  $e^{-1/8t}$  since  $B(t) \sim \mathcal{N}(0, t)$ . Overall, we get

$$\mathbf{Pr}[\neg \mathcal{E}] \leq \sum_{j=1}^n \mathbf{Pr}[\mathcal{D}^{(j)}] \leq \sum_{j=1}^n \left( \mathbf{Pr}[M'(\sigma_j^2) < -1/2] + \mathbf{Pr}[M(\sigma_j^2) > 1/2] \right) \leq 2n \cdot e^{-1/8\sigma^2}$$

Next, we bound  $|\mathbf{E}_{Z^{(i+1)}}[f(\text{trnc}(H_i + p \cdot Z^{(i+1)})) - f(\text{trnc}(H_i))]|$  conditioned on the event  $\mathcal{E}_i$ , for  $i = 0, 1, \dots, m-1$ . Let  $v = H_i$ . Condition on the event  $\mathcal{E}_i$ , and in fact condition on the entire history in the first  $i$  steps, which in particular fixes  $v$ . By Claim 19, we have

$$\begin{aligned} \left| \mathbf{E}_{Z^{(i+1)}} [f(v + p \cdot Z^{(i+1)}) - f(v)] \right| &= \left| \mathbf{E}_{Z^{(i+1)}} \mathbf{E}_{\rho} [f_{\rho}(2p \cdot Z^{(i+1)}) - f_{\rho}(\vec{0})] \right| \\ &\leq \mathbf{E}_{\rho} \left| \mathbf{E}_{Z^{(i+1)}} [f_{\rho}(2p \cdot Z^{(i+1)}) - f_{\rho}(\vec{0})] \right|. \end{aligned}$$

By Claim 20 we have that the latter is at most  $(2p)^2 t \delta + O(p^4 n^4 \delta^2)$  as long as  $2p \leq 1/2n$ . We wish to show a similar bound on the truncated version of  $H_i + p \cdot Z^{(i+1)}$ . Note that conditioned on  $\mathcal{E}_i$ , we have  $H_i = \text{trnc}(H_i)$ , but this is not necessarily the case for  $H_i + p \cdot Z^{(i+1)}$ . Using Claim 17 we get  $|\mathbf{E}_{Z^{(i+1)}}[f(p \cdot Z^{(i+1)} + v) - f(\text{trnc}(p \cdot Z^{(i+1)} + v))]| \leq 2 \cdot \sum_{k=1}^{\infty} e^{-k/(16p^2 n)} \cdot n^k$ . By the triangle inequality we get

$$\begin{aligned} &\left| \mathbf{E}_{Z^{(i+1)}} [f(\text{trnc}(v + p \cdot Z^{(i+1)})) - f(\text{trnc}(v))] \right| \\ &\leq \left| \mathbf{E}_{Z^{(i+1)}} [f(\text{trnc}(v + p \cdot Z^{(i+1)})) - f(v + p \cdot Z^{(i+1)})] \right| \\ &\quad + \left| \mathbf{E}_{Z^{(i+1)}} [f(v + p \cdot Z^{(i+1)}) - f(\text{trnc}(v))] \right| \\ &\leq \left( 2 \cdot \sum_{k=1}^{\infty} e^{-k/(16p^2 n)} \cdot n^k \right) + \left( 4p^2 \cdot \delta \cdot t + O(p^4 n^4 \delta^2) \right). \end{aligned} \quad (3)$$

To finish the proof, using triangle inequality we have

$$|\mathbf{E}[f(\text{trnc}(H_m)) - f(\vec{0})]| \leq |\mathbf{E}[f(\text{trnc}(H_m)) \cdot \mathbb{1}_{\mathcal{E}} - f(\vec{0})]| + |\mathbf{E}[f(\text{trnc}(H_m)) \cdot \mathbb{1}_{\neg \mathcal{E}}]|$$

We bound the second summand by  $\mathbf{Pr}[\neg \mathcal{E}]$  since  $f$  is bounded in  $[-1, 1]$  on truncated vectors, whereas the first summand is bounded using a telescopic sum of the  $m+1$  hybrids:

$$\begin{aligned} &|\mathbf{E}[f(\text{trnc}(H_m)) \cdot \mathbb{1}_{\mathcal{E}} - f(\vec{0})]| \\ &\leq \sum_{i=0}^{m-1} |\mathbf{E}[f(\text{trnc}(H_{i+1})) \cdot \mathbb{1}_{\mathcal{E}_1 \wedge \dots \wedge \mathcal{E}_{i+1}} - f(\text{trnc}(H_i)) \cdot \mathbb{1}_{\mathcal{E}_1 \wedge \dots \wedge \mathcal{E}_i}]| \\ &\leq \sum_{i=0}^{m-1} |\mathbf{E}[f(\text{trnc}(H_{i+1})) \cdot \mathbb{1}_{\mathcal{E}_1 \wedge \dots \wedge \mathcal{E}_i} - f(\text{trnc}(H_i)) \cdot \mathbb{1}_{\mathcal{E}_1 \wedge \dots \wedge \mathcal{E}_i}]| \\ &\quad + |\mathbf{E}[f(\text{trnc}(H_{i+1})) \cdot (\mathbb{1}_{\mathcal{E}_1 \wedge \dots \wedge \mathcal{E}_{i+1}} - \mathbb{1}_{\mathcal{E}_1 \wedge \dots \wedge \mathcal{E}_i})]| \\ &\leq \sum_{i=0}^{m-1} \left( 2 \cdot \sum_{k=1}^{\infty} e^{-k/(16p^2 n)} \cdot n^k \right) + \left( 4p^2 \cdot \delta \cdot t + O(p^4 n^4 \delta^2) \right) + \mathbf{E}[\mathbb{1}_{\mathcal{E}_1 \wedge \dots \wedge \mathcal{E}_{i+1}} - \mathbb{1}_{\mathcal{E}_1 \wedge \dots \wedge \mathcal{E}_i}] \\ &\hspace{15em} (\text{Eq. (3), } f \text{ is bounded}) \\ &\leq m \cdot \left( 2 \cdot \sum_{k=1}^{\infty} e^{-k/(16p^2 n)} \cdot n^k \right) + 4p^2 \cdot \delta \cdot t + O(p^4 n^4 \delta^2) + \mathbf{Pr}[\neg \mathcal{E}]. \end{aligned}$$

Overall,

$$\begin{aligned} |\mathbf{E}[f(\text{trnc}(H_m)) - f(\vec{0})]| &\leq m \cdot \left( 2 \cdot \sum_{k=1}^{\infty} e^{-k/(16p^2 n)} \cdot n^k \right) + 4p^2 \cdot \delta \cdot t + O(p^4 n^4 \delta^2) + 2 \mathbf{Pr}[\neg \mathcal{E}] \\ &= m \cdot \left( 2 \cdot \sum_{k=1}^{\infty} e^{-km/(16n)} \cdot n^k \right) + 4m^{-1} \cdot \delta \cdot t + O(m^{-2} n^4 \delta^2) + 2 \mathbf{Pr}[\neg \mathcal{E}] \end{aligned}$$

Taking  $m \rightarrow \infty$  gives the upper bound  $4\delta \cdot t + 2 \mathbf{Pr}[\neg \mathcal{E}] \leq 4\delta \cdot t + 4n \cdot e^{-1/8\sigma^2}$  as promised.  $\blacktriangleleft$



# Classical Algorithms from Quantum and Arthur-Merlin Communication Protocols

Lijie Chen

Massachusetts Institute of Technology, Cambridge, MA, USA

lijieche@mit.edu

Ruosong Wang

Carnegie Mellon University, Pittsburgh, PA, USA

ruosongw@andrew.cmu.edu

---

## Abstract

In recent years, the polynomial method from circuit complexity has been applied to several fundamental problems and obtains the state-of-the-art running times (e.g., R. Williams's  $n^3/2^{\Omega(\sqrt{\log n})}$  time algorithm for APSP). As observed in [Alman and Williams, STOC 2017], almost all applications of the polynomial method in algorithm design ultimately rely on certain (probabilistic) low-rank decompositions of the computation matrices corresponding to key subroutines. They suggest that making use of low-rank decompositions directly could lead to more powerful algorithms, as the polynomial method is just one way to derive such a decomposition.

Inspired by their observation, in this paper, we study another way of *systematically constructing low-rank decompositions of matrices* which could be used by algorithms – *communication protocols*. Since their introduction, it is known that various types of communication protocols lead to certain low-rank decompositions (e.g., P protocols/rank, BQP protocols/approximate rank). These are usually interpreted as approaches for proving communication lower bounds, while in this work we explore the other direction.

We have the following two generic algorithmic applications of communication protocols:

- **Quantum Communication Protocols and Deterministic Approximate Counting.** Our first connection is that a fast BQP communication protocol for a function  $f$  implies a fast deterministic additive approximate counting algorithm for a related pair counting problem. Applying known BQP communication protocols, we get fast deterministic additive approximate counting algorithms for Count-OV (#OV), Sparse Count-OV and Formula of SYM circuits. In particular, our approximate counting algorithm for #OV runs in near-linear time for all dimensions  $d = o(\log^2 n)$ . Previously, even no truly-subquadratic time algorithm was known for  $d = \omega(\log n)$ .
- **Arthur-Merlin Communication Protocols and Faster Satisfying-Pair Algorithms.** Our second connection is that a fast  $\text{AM}^{\text{cc}}$  protocol for a function  $f$  implies a faster-than-bruteforce algorithm for  $f$ -Satisfying-Pair. Using the classical Goldwasser-Sispe AM protocols for approximating set size, we obtain a new algorithm for approximate Max-IP $_{n,c \log n}$  in time  $n^{2-1/O(\log c)}$ , matching the state-of-the-art algorithms in [Chen, CCC 2018].

We also apply our second connection to shed some light on long-standing open problems in communication complexity. We show that if the Longest Common Subsequence (LCS) problem admits a fast (computationally efficient)  $\text{AM}^{\text{cc}}$  protocol (polylog( $n$ ) complexity), then polynomial-size Formula-SAT admits a  $2^{n-n^{1-\delta}}$  time algorithm for any constant  $\delta > 0$ , which is conjectured to be unlikely by a recent work [Abboud and Bringmann, ICALP 2018]. The same holds even for a fast (computationally efficient)  $\text{PH}^{\text{cc}}$  protocol.

**2012 ACM Subject Classification** Theory of computation → Communication complexity

**Keywords and phrases** Quantum communication protocols, Arthur-Merlin communication protocols, approximate counting, approximate rank



© Lijie Chen and Ruosong Wang;  
licensed under Creative Commons License CC-BY  
10th Innovations in Theoretical Computer Science (ITCS 2019).  
Editor: Avrim Blum; Article No. 23; pp. 23:1–23:20



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

Digital Object Identifier 10.4230/LIPIcs.ITCS.2019.23

**Related Version** A full version of the paper is available at <https://arxiv.org/abs/1811.07515>.

**Acknowledgements** The first author is grateful to Josh Alman, Chi-Ning Chou, Mika Göös, and Ryan Williams for helpful discussions during this work. We are grateful to anonymous reviewers for many helpful and inspiring comments on this paper. In particular, we thank one anonymous reviewer for pointing out that Theorem 5 can be improved using the polynomial method.

## 1 Introduction

Recent works have shown that the polynomial method, a classical technique for proving circuit lower bounds [41, 45], can be useful in designing efficient algorithms [48, 50, 6, 10, 8, 36, 7].

At a very high level, these algorithms proceed as follows: (1) identify a key subroutine of the core algorithm which has a certain low-degree polynomial representation; (2) replace that subroutine by the corresponding polynomials, and reduce the whole problem to a certain *batched evaluation problem of sparse polynomials*; (3) embed that polynomial evaluation problem to *multiplication of two low-rank (rectangular) matrices*, and apply the fast rectangular matrix multiplication algorithm [26].

As [9] point out. In term of step (3), these algorithms are ultimately making use of the fact that the corresponding matrices of some circuits or subroutines have low *probabilistic rank*. [9] suggest that the *probabilistic rank*, or various low-rank decompositions of matrices in general<sup>1</sup>, could be more powerful than the polynomial method, and lead to more efficient algorithms, as the polynomial method is just one way to construct them.

It has been noted for a long time that communication protocols are closely related to various notions of rank of matrices. To list a few: deterministic communication complexity is lower bounded by the logarithm of the *rank* of the matrix [37]; quantum communication complexity is lower bounded by the logarithm of the *approximate rank* of the matrix [16, 19]; UPP communication complexity is equivalent to the logarithm of the *sign-rank* of the matrix [40].

These connections are introduced (and usually interpreted) as methods for proving communication complexity lower bounds (see, e.g. the survey by Lee and Shraibman [35]), but they can also be interpreted in the other direction, as a way to *systematically construct low-rank decompositions of matrices*.

In this paper, we explore the connection between different types of communication protocols and low-rank decompositions of matrices and establish several applications in algorithm design. For all these connections, we start with an efficient communication protocol for a problem  $F$ , which implies an efficiently constructible low-rank decomposition of the corresponding communication matrix of  $F$ , from which we can obtain fast algorithms.

In fact, in our applications of quantum communication protocols, we also consider  $k$ -party protocols, and our algorithms rely on the approximate low-rank decomposition of the tensor of the corresponding communication problem. To the best of our knowledge, this is the first time that *approximate tensor rank* is used in algorithm design (approximate rank has been used before, see e.g. [11, 18, 13, 12] and the corresponding related works section).<sup>2</sup>

<sup>1</sup> A low probabilistic rank implies a probabilistic low-rank decomposition of the matrix.

<sup>2</sup> We remark that a concurrent work [52] makes algorithmic use of *non-negative tensor approximate rank* to construct an optimal data structure for the succinct rank problem.



## 1.1 Quantum Communication Protocols and Deterministic Approximate Counting

Our first result is a generic connection between quantum communication protocols and deterministic approximate counting algorithms.

► **Theorem 1.** (Informal) Let  $\mathcal{X}, \mathcal{Y}$  be finite sets and  $f : \mathcal{X} \times \mathcal{Y} \rightarrow \{0, 1\}$  be a Boolean function. Suppose  $f$  has a quantum communication protocol  $\mathcal{P}^3$  with complexity  $C(\mathcal{P})$  and error  $\varepsilon$ . Then there is a classical deterministic algorithm  $\mathcal{C}$  that receives  $A \subseteq \mathcal{X}, B \subseteq \mathcal{Y}$  as input, and outputs a number  $E$  such that

$$\left| \sum_{(x,y) \in A \times B} f(x,y) - E \right| \leq \varepsilon \cdot |A| \cdot |B|.$$

Furthermore,  $\mathcal{C}$  runs in  $(|A| + |B|) \cdot 2^{O(C(\mathcal{P}))}$  time.

We remark here that there is a simple randomized algorithm running in sub-linear time via random-sampling. Thus the above algorithm is indeed a derandomization of that randomized algorithm.

The above theorem can also be easily generalized to the (number-in-hand)  $k$ -party case. See Section 2.5 for the definition of the multiparty quantum communication model.

► **Theorem 2.** (Informal) Let  $\mathcal{X}_1, \mathcal{X}_2, \dots, \mathcal{X}_k$  be finite sets and  $f : \mathcal{X}_1, \mathcal{X}_2, \dots, \mathcal{X}_k \rightarrow \{0, 1\}$  be a Boolean function. Suppose  $f$  has a  $k$ -party quantum communication protocol  $\mathcal{P}$  with complexity  $C(\mathcal{P})$  and error  $\varepsilon$ . Then there is a classical deterministic algorithm  $\mathcal{C}$  that receives  $X_1 \subseteq \mathcal{X}_1, X_2 \subseteq \mathcal{X}_2, \dots, X_k \subseteq \mathcal{X}_k$  as input, and outputs a number  $E$  such that

$$\left| \sum_{x_1 \in X_1, x_2 \in X_2, \dots, x_k \in X_k} f(x_1, x_2, \dots, x_k) - E \right| \leq \varepsilon \cdot \prod_{i=1}^k |X_i|.$$

Furthermore,  $\mathcal{C}$  runs in  $(|X_1| + |X_2| + \dots + |X_k|) \cdot 2^{O(C(\mathcal{P}))}$  time.

### Sketching Algorithms

In fact, Theorem 2 implies a stronger *sketching algorithm*. Given subsets  $X_1, X_2, \dots, X_k$ , the algorithm first computes a  $w = 2^{O(C(\mathcal{P}))}$  size sketch  $\text{sk}_i$  from each  $X_i$  in  $O(|X_i| \cdot w)$  time deterministically, and the number  $E$  can be computed from these  $\text{sk}_i$ 's in  $O(k \cdot w)$  time.

The sketch computed by the algorithm is in fact a vector in  $\mathbb{R}^w$ , and it satisfies a nice additive property. That is, the sketch of  $X_1 \sqcup X_2$  (union as a multi-set) is simply  $\text{sk}(X_1) + \text{sk}(X_2)$ .

Applying existing quantum communication protocols, we obtain several applications of Theorem 1 and Theorem 2.

#### 1.1.1 Set-Disjointness and Approximate #OV and # $k$ -OV

We first consider the famous SET-DISJOINTNESS problem (Alice and Bob get two vectors  $u$  and  $v$  in  $\{0, 1\}^d$  correspondingly, and want to determine whether  $\langle u, v \rangle = 0$ ), which has an efficient quantum communication protocol [1] with communication complexity  $O(\sqrt{d})$ .

<sup>3</sup> We need some technical condition on  $\mathcal{P}$ , see Corollary 29 for details.

The corresponding count problem for SET-DISJOINTNESS is the counting version of the Orthogonal Vectors problem (OV), denoted as  $\#OV_{n,d}$ . In this problem, we are given two sets of  $n$  vectors  $S, T \subseteq \{0, 1\}^d$ , and the goal is to count the number of pairs  $u \in S, v \in T$  such that  $\langle u, v \rangle = 0$ .

Applying the quantum communication protocol for SET-DISJOINTNESS and Theorem 2, we immediately get an algorithm for the approximate version of  $\#OV$ .

► **Theorem 3.** *For any  $d$  and any constant  $\varepsilon > 0$ ,  $\#OV_{n,d}$  can be approximated deterministically with additive error  $\varepsilon \cdot n^2$  in  $n \cdot 2^{O(\sqrt{d})}$  time. In particular, it runs in  $n^{1+o(1)}$  time when  $d = o(\log^2 n)$ .*

### Comparison with [22]

[22] gives a *deterministic exact counting* algorithm for  $\#OV_{n,c \log n}$ , which runs in  $n^{2-O(1/\log c)}$  time. Note that their running time is  $n^{2-o(1)}$  when  $d = \omega(\log n)$ , while our algorithm only achieves an additive approximation, but runs in *near-linear* time for all  $d = o(\log^2 n)$ .

Another closely related problem, COUNTING PARTIAL MATCH, is the problem that given  $n$  query strings from  $\{0, 1, \star\}^d$  ( $\star$  is a “don’t care”) and  $n$  strings from  $\{0, 1\}^d$ , and the goal is to count the number of matching string and query pairs.

Using known reductions between PARTIAL MATCH and OV (see, e.g., Section 2 in [6]), together with the approximate counting algorithm for  $\#OV$ , we can also solve COUNTING PARTIAL MATCH approximately in the same running time.

The approximate counting algorithm for  $\#OV$  can be easily generalized to solve  $\#k$ -OV, which is the problem that given  $k$  sets of  $n$  vectors  $X_1, X_2, \dots, X_k \subseteq \{0, 1\}^d$ , and count the number of  $k$ -tuples  $u_1 \in X_1, u_2 \in X_2, \dots, u_k \in X_k$  such that  $\langle u_1, u_2, \dots, u_k \rangle = 0$ .<sup>4</sup>

Applying Theorem 2 and observe that the 2-party SET-DISJOINTNESS protocol in [1] can be easily generalized to solve the  $k$ -party case (in  $k$ -party SET-DISJOINTNESS, there are  $k$  players getting  $u_1, u_2, \dots, u_k$  respectively, and they want to determine whether  $\langle u_1, u_2, \dots, u_k \rangle = 0$ ), we obtain the following approximate counting algorithm for  $\#k$ -OV.

► **Theorem 4.** *For any integers  $k, d$  and any constant  $\varepsilon > 0$ ,  $\#k$ -OV $_{n,d}$  can be approximated deterministically with additive error  $\varepsilon \cdot n^k$  in  $n \cdot 2^{O(k\sqrt{d})}$  time. In particular, it runs in  $n^{1+o(1)}$  time when  $k$  is a constant and  $d = o(\log^2 n)$ .*

► **Remark.** We remark that similar algorithms with slightly worse running time ( $n \cdot d^{O(\sqrt{d})}$  time for additive approximation to  $\#OV_{n,d}$ ) can also be derived using the polynomial method. However, we think our new algorithms via quantum communication protocols have the following extra benefits: (1) our algorithm is slightly faster, with a running time of  $n \cdot 2^{O(\sqrt{d})}$ ; (2) our algorithm is derived via a general connection. Once the connection is set up, the algorithm follows in an elegant and black-box way. We hope this general connection could stimulate more applications of quantum communication protocols.

### 1.1.2 Sparse Set-Disjointness and Approximate Sparse $\#OV$

Next we consider a sparse version of SET-DISJOINTNESS, in which Alice and Bob get two sparse vectors  $u, v \in \{0, 1\}_{\leq d}^m$ <sup>5</sup>, and want to decide whether  $\langle u, v \rangle = 0$ .

<sup>4</sup> the generalized inner product of  $k$  vectors, is defined as  $\langle u_1, u_2, \dots, u_k \rangle = \sum_{i=1}^d \prod_{j=1}^k (u_j)_i$ .

<sup>5</sup> We use  $\{0, 1\}_{\leq d}^m$  to denote all Boolean vectors of length  $m$  with at most  $d$  ones.

Using the famous quantum-walk algorithm for ELEMENT DISTINCTNESS [14], there is an  $O(d^{2/3} \log m)$  communication protocol for sparse SET-DISJOINTNESS, which is much better than the  $O(\sqrt{m})$  protocol for SET-DISJOINTNESS when  $m \gg d$ .

Applying this protocol and Theorem 1, we can give an algorithm for a sparse version of #OV, denoted as #Sparse-OV $_{n,m,d}$ , in which we are given sets  $A, B \subseteq \{0, 1\}_{\leq d}^m$  of  $n$  vectors, and the goal is to count the number of distinct  $(a, b) \in A \times B$  such that  $\langle a, b \rangle = 0$ . Formally, we have:

► **Theorem 5.** *For integers  $n, m, d$  and any constant  $\varepsilon > 0$ , #Sparse-OV $_{n,m,d}$  can be approximated deterministically with additive error  $\varepsilon \cdot n^2$  in*

$$n \cdot 2^{O(d^{2/3} \log(m))}$$

*time. In particular, when  $m = \text{poly}(d)$  and  $d = o\left(\left(\frac{\log n}{\log \log n}\right)^{1.5}\right)$ , it runs in  $n^{1+o(1)}$  time.*

We remark that it is possible to improve Theorem 5 via the polynomial method. Again, we emphasize that our focus here is to provide direct applications of our general framework, with the hope that it could stimulate more applications of quantum communication protocols in the classical settings.

### 1.1.3 Approximate Counting for Formula $\circ$ SYM Circuits

Finally, we apply our algorithm to approximately count solutions (i.e., satisfying assignments) to a class of circuits, for which no non-trivial algorithms were previously known.

A Formula  $\circ$  SYM circuit of size  $m$  is a formula with {AND, OR, NOT} basis on  $m$  SYM gates<sup>6</sup> at the bottom. Using the quantum query algorithm for FORMULA EVALUATION [15] and the split-and-list technique, we obtain the following deterministic approximate counting algorithm for Formula  $\circ$  SYM circuits:

► **Theorem 6.** *For any constant  $\varepsilon > 0$ , the number of solutions to a Formula  $\circ$  SYM circuit of size  $m$  can be approximated deterministically within  $\varepsilon \cdot 2^n$  additive error in  $2^{O(n^{1/2} m^{1/4+o(1)} \sqrt{\log n + \log m})}$  time. In particular, when  $m = n^{2-\delta}$  for some  $\delta > 0$ , the running time is  $2^{o(n)}$ .*

Previously, even no non-trivial deterministic approximate counting algorithms for AND  $\circ$  SYM circuits were known. A recent line of works [31, 32, 44], culminating in [39], construct a PRG for AND $_m$   $\circ$  THR circuits with seed length  $\text{poly}(\log m, \delta^{-1}) \cdot \log n$ , using which one can obtain a quasi-polynomial time deterministic approximate counting algorithm for polynomial size AND  $\circ$  THR circuits. However, their PRG constructions rely on the fact that the solution set of an AND $_m$   $\circ$  THR circuit is a *polytope*, while the solution set of an AND  $\circ$  SYM circuit may not have such a nice geometric structure.

In fact, the only property we need for SYM gates is that they admit an efficient *classical*  $k$ -party communication protocol when the inputs are divided to  $k$  players (each player sends the contribution of her part). Our algorithm actually works for the following more general problem.

► **Problem 1.** *Given  $k$  sets of  $n$  vectors  $X_1, X_2, \dots, X_k \subseteq \{0, \dots, r\}^d$  and  $d$  functions  $f_1, f_2, \dots, f_d$  where each  $f_i$  is from  $[r]^k$  to  $\{0, 1\}$ , and a Boolean formula  $\mathcal{F} : \{0, 1\}^d \rightarrow \{0, 1\}$  of  $O(1)$  fan-in. Count the number of  $k$ -tuples  $u_1 \in X_1, u_2 \in X_2, \dots, u_k \in X_k$  such that*

$$\mathcal{F}(f_1(u_{1,1}, u_{2,1}, \dots, u_{k,1}), f_2(u_{1,2}, u_{2,2}, \dots, u_{k,2}), \dots, f_d(u_{1,d}, u_{2,d}, \dots, u_{k,d})) = 1.$$

<sup>6</sup> A SYM gate is a gate whose output only depends on the number of ones in the input.

► **Theorem 7.** *For any constant  $\varepsilon > 0$ , the above problem can be solved deterministically in  $n \cdot 2^{O(d^{1/2+o(1)} \cdot k(\log d + \log r))}$  time, within  $\varepsilon \cdot n^k$  additive error.*

## 1.2 Arthur-Merlin Communication Protocols and a New Approximate Max-IP Algorithm

Our second connection is an algorithmic application of  $\text{AM}^{\text{cc}}$  protocols. We first define  $\text{AM}^{\text{cc}}$  protocols formally.

► **Definition 8.** An Arthur-Merlin communication protocol ( $\text{AM}^{\text{cc}}$ )  $\Pi$  for a partial function  $F : \mathcal{X} \times \mathcal{Y} \rightarrow \{0, 1, \perp\}$ <sup>7</sup> proceeds as follows:

- Alice holds input  $x \in \mathcal{X}$  and Bob holds input  $y \in \mathcal{Y}$ .
- Alice and Bob toss some public coins jointly and send the random string  $r \in \{0, 1\}^*$  to Merlin ( $r$  is called the random challenge).
- Based on  $x$ ,  $y$  and the random challenge  $r$ , Merlin sends Alice and Bob a proof  $\psi$ , and Alice and Bob decide to accept or not independently and deterministically. We require the following conditions:
  - If  $F(x, y) = 1$ , with probability  $1 - \varepsilon$  over the random challenge  $r$ , there is a proof  $\psi$  from Merlin such that Alice and Bob both accept.
  - If  $F(x, y) = 0$ , with probability  $1 - \varepsilon$  over the random challenge  $r$ , there is no proof  $\psi$  from Merlin such that Alice and Bob both accept.

We call the parameter  $\varepsilon$  the error of the protocol  $\Pi$ . Moreover, we say the protocol is *computationally efficient* if Alice and Bob's behavior can be computed in polynomial-time w.r.t. their input lengths.

We show that for any function  $F$ , a low-complexity and computationally efficient  $\text{AM}^{\text{cc}}$  protocol implies a faster algorithm for the corresponding  $F$ -Satisfying-Pair problem (defined below).

For a partial function  $F : \mathcal{X} \times \mathcal{Y} \rightarrow \{0, 1, \perp\}$ , where  $\mathcal{X}$  and  $\mathcal{Y}$  are two sets, we define  $F$ -Satisfying-Pair <sub>$n$</sub>  as the problem that given two sets  $A \subseteq \mathcal{X}$  and  $B \subseteq \mathcal{Y}$  of size  $n$ , distinguish between the following two cases: (1) There is an  $(x, y) \in A \times B$  such that  $F(x, y) = 1$ . (2) For all  $(x, y) \in A \times B$ ,  $F(x, y) = 0$ .

► **Theorem 9 (Algorithms from  $\text{AM}^{\text{cc}}$  protocols).** *Let  $F : \mathcal{X} \times \mathcal{Y} \rightarrow \{0, 1, \perp\}$  be a partial function. Suppose there is a computationally efficient  $\text{AM}^{\text{cc}}$  protocol for  $F$  with communication complexity  $T$  and error  $\varepsilon$ . Then for  $n$  such that  $2^T \leq (\sqrt{\varepsilon}n)^{0.1}$ , there is an  $O(\varepsilon n^2 \cdot \text{polylog}(n) + n \cdot 2^T)$  time randomized algorithm for  $F$ -Satisfying-Pair <sub>$n$</sub> .*

### 1.2.1 A New Algorithm for Approximate Max-IP

The first application of Theorem 9 is a new algorithm for approximate Maximum Inner Product. We use  $\text{Max-IP}_{n,d}$  to denote the problem that given sets  $A, B \subseteq \{0, 1\}^d$  with size  $n$ , compute  $\text{Max}(A, B) := \max_{(a,b) \in A \times B} \langle a \cdot b \rangle$ .

To phrase this as an  $F$ -Satisfying-Pair problem, we first define the following gap inner product problem.

<sup>7</sup>  $F(x, y) = \perp$  means  $F(x, y)$  is undefined.

► **Definition 10** (Multiplicative-Gap Inner Product). Consider the following problem, denoted as  $\text{Gap-Inner-Product}_d$ , Alice and Bob hold strings  $x, y \in \{0, 1\}^d$  respectively, and they are given an integer  $\tau$ . They want to distinguish between the following two cases: (Yes)  $x \cdot y \geq 2\tau$ ; (No)  $x \cdot y \leq \tau$ .

Adapting the classical Goldwasser-Sisler AM protocol for approximating set size [28], we can derive an efficient  $\text{AM}^{\text{cc}}$  protocol for  $\text{Gap-Inner-Product}_d$ .

► **Lemma 11** ( $\text{AM}^{\text{cc}}$  protocol for  $\text{Gap-Inner-Product}_d$ ). *There is an  $\text{AM}^{\text{cc}}$  protocol which solves  $\text{Gap-Inner-Product}_d$  with error  $\varepsilon$  and communication complexity  $\log \binom{d}{\leq O(\log \varepsilon^{-1})}$ .*<sup>8</sup>

Applying Theorem 9, the following algorithm for approximating Max-IP follows directly, matching the previous best algorithm in [23].

► **Corollary 12.** *There is an algorithm for computing a 2-approximation to  $\text{Max-IP}_{n, c \log n}$ , which runs in  $n^{2-1/O(\log c)}$  time.*

► **Remark.** The constant 2 in Corollary 12 can be replaced by any other constant  $\kappa > 1$ .

We remark here that a direct application of the Goldwasser-Sisler protocol and parallel repetition leads to a communication protocol with communication complexity  $O(\log d \log \varepsilon^{-1})$ , which is slightly worse than Lemma 11. In particular, such a protocol only gives an algorithm with running time  $n^{2-1/O(\log d)}$ , which is worse than  $n^{2-1/O(\log c)}$  when  $c \ll d = c \log n$ . In order to get the improved complexity in Lemma 11, we make use of a clever sampling scheme using Poisson distributions, see Section 4.1 for details.

## 1.2.2 Evidences that Longest Common Subsequence and Edit Distance do not Have Fast $\text{AM}^{\text{cc}}$ Protocols

It has been a long-standing open problem in communication complexity to prove an  $\omega(\log n)$   $\text{AM}^{\text{cc}}$  lower bound for any explicit function [17, 29, 30]—it is consistent with our current knowledge that all known natural communication problems have  $O(\log n)$   $\text{AM}^{\text{cc}}$  protocols.

We consider two natural communication problems here,  $\text{LCS}_d^{\text{cc}}$  and  $\text{Edit-Dist}_d^{\text{cc}}$ , in which Alice and Bob hold strings  $x, y \in \{0, 1\}^d$  respectively, and are given an integer  $\tau$ . Their goal is to decide whether  $\text{LCS}(x, y) \geq \tau$  ( $\text{Edit-Distance}(x, y) \geq \tau$ ).

Our Theorem 9 shows that if  $\text{LCS}^{\text{cc}}$  or  $\text{Edit-Dist}^{\text{cc}}$  admit low-complexity and computationally efficient  $\text{AM}^{\text{cc}}$  protocols, it would imply non-trivial algorithms for the corresponding  $F$ -Satisfying-Pair problem. By a known reduction in [3], that would, in turn, implies non-trivial algorithms for  $\text{Formula-SAT}^9$ —much faster than the current state-of-the-art [47]! Therefore, at least for these two problems, constructing low-complexity  $\text{AM}^{\text{cc}}$  protocol could be hard, which may also be viewed as an evidence that they do not have efficient  $\text{AM}^{\text{cc}}$  protocols.

► **Theorem 13.** *If  $\text{LCS}_d^{\text{cc}}$  admits computationally efficient  $\text{AM}^{\text{cc}}$  protocols with complexity  $\text{polylog}(d)$ , then  $\text{Formula-SAT}$  of polynomial-size formulas admits an  $2^{n-n^{1-\delta}}$  time algorithm for any constant  $\delta > 0$ . The same holds for  $\text{Edit-Dist}^{\text{cc}}$  in place of  $\text{LCS}^{\text{cc}}$ .*

The state-of-the-art algorithm for  $\text{Formula-SAT}$  runs in  $o(2^n)$  time only when the formula size is smaller than  $n^3$  [47]. It is even purposed as a hypothesis that no  $2^n/n^{\omega(1)}$  time algorithm exists for  $n^{3+\Omega(1)}$ -size  $\text{Formula-SAT}$  in [2]. Therefore, our results imply that if  $\text{LCS}^{\text{cc}}$  or  $\text{Edit-Dist}^{\text{cc}}$  admits fast (computationally efficient)  $\text{AM}^{\text{cc}}$  protocols, then that would refute the hypothesis in [2]:

<sup>8</sup>  $\binom{n}{\leq m}$  denotes  $\sum_{i=0}^m \binom{n}{i}$ .

<sup>9</sup>  $\text{Formula-SAT}$  is the problem that deciding whether a given formula is satisfiable.

► **Corollary 14.** *Under the following hypothesis<sup>10</sup>,  $LCS_d^{cc}$  and  $Edit-Dist_d^{cc}$  do not admit computationally efficient  $AM^{cc}$  protocols with complexity  $\text{polylog}(d)$ :*

- *There is a constant  $\delta > 0$  such that Formula-SAT of polynomial-size formulas requires  $2^{n-n^{1-\delta}}$  time.*

In the full version of this paper, we show that the above corollary can be generalized to hold for computationally efficient  $PH^{cc}$  protocols (see the full version for a formal definition). Formally, we have:

► **Theorem 15.** *Under the same hypothesis as in Corollary 14,  $LCS_d^{cc}$  and  $Edit-Dist_d^{cc}$  do not admit computationally efficient  $PH^{cc}$  protocols with complexity  $\text{polylog}(d)$ .*

## 1.3 Related Works

### 1.3.1 Communication Protocols and Fine-Grained Complexity

Recently, since the breakthrough work of [5], communication protocols have been applied to *fine-grained complexity*, and several tight conditional hardness results are proved for many fundamental approximate problems in P [5, 33, 4, 23, 24, 25, 43].

Among these works, the most related one is [23], in which the author also makes use of the  $BQP^{cc}$  protocol for SET-DISJOINTNESS for a different purpose. In [23], the  $BQP^{cc}$  protocol is used to establish a *reduction* from OV to approximate  $\{-1, 1\}$ -Max-IP<sup>11</sup>, thereby showing the SETH-hardness of approximating  $\{-1, 1\}$ -Max-IP. On the other hand, in this work we use  $BQP^{cc}$  protocols directly for algorithmic purposes.

### 1.3.2 Other Algorithmic Applications of Approximate Rank

Alon studies the approximate rank of the identity matrix  $I_n$  in [11]. It is shown that it is at least  $\Omega\left(\frac{\log n}{\varepsilon^2 \log(1/\varepsilon)}\right)$  and at most  $O\left(\frac{\log n}{\varepsilon^2}\right)$ . Built upon this result, several applications in geometry, coding theory, extremal finite set theory and the study of sample spaces supporting nearly independent random variables are derived. The lower bound also has applications in combinatorial geometry and in the study of locally correctable codes over real and complex numbers, as shown in [18]. In [13, 12], several bounds on approximate rank are derived, together with applications of approximate rank in approximating Nash Equilibria, approximating densest bipartite subgraph and covering convex bodies.

## 2 Preliminaries

### 2.1 Fast Rectangular Matrix Multiplication

Similar to previous algorithms using the polynomial method (see, e.g., [50, 10, 6]), our algorithms also make use of algorithms for fast rectangular matrix multiplication.

► **Theorem 16** ([26, 27]). *There is an  $N^2 \cdot \text{polylog}(N)$  time algorithm for multiplying two matrices  $A$  and  $B$  with size  $N \times N^\alpha$  and  $N^\alpha \times N$ , where  $\alpha > 0.172$ .*

<sup>10</sup> which is much weaker than the hypothesis in [2]

<sup>11</sup> a variant of Max-IP with vectors in  $\{-1, 1\}^d$  instead of  $\{0, 1\}^d$

## 2.2 Random Variables and Poisson Distributions

Throughout the paper, we use  $X \simeq Y$  to mean that  $X$  and  $Y$  have the same distribution. We use  $X \succeq Y$  to denote stochastic dominance, i.e.,  $X \succeq Y$  iff for any  $t \in \mathbb{R}$ ,  $\Pr[X \geq t] \geq \Pr[Y \geq t]$ .

We use  $\text{Pois}(\lambda)$  to denote a Poisson distribution with parameter  $\lambda$ . We will need the following two facts about Poisson distributions. The proof can be found in the full version.

► **Lemma 17.** *Suppose  $\{X_i\}_{i=1}^n$  is a set of independent random variables with  $X_i \sim \text{Pois}(\lambda_i)$ , then  $\sum_{i=1}^n X_i \sim \text{Pois}(\sum_{i=1}^n \lambda_i)$ .*

► **Lemma 18.**  $\Pr[\text{Pois}(\lambda) \geq 1.2\lambda] \leq e^{-0.01\lambda}$  and  $\Pr[\text{Pois}(\lambda) \leq 0.8\lambda] \leq e^{-0.01\lambda}$ .

## 2.3 Tensor Ranks

In this paper we are interested in the approximate tensor rank with respect to the  $\ell_\infty$  norm. For more on approximate tensor rank with respect to other norms and their applications, see [46] and the references therein. Now we introduce some relevant definitions.

► **Definition 19.** We say a tensor  $T \in \mathbb{R}^{n_1 \times n_2 \times \dots \times n_k}$  is *simple* if  $T = v_1 \otimes v_2 \otimes \dots \otimes v_k$  where  $v_i \in \mathbb{R}^{n_i}$ .

► **Definition 20.** For a tensor  $T \in \mathbb{R}^{n_1 \times n_2 \times \dots \times n_k}$ , its  $\text{rank}(T)$  is defined to be the smallest integer  $r$  such that  $T = \sum_{i=1}^r A_i$  and  $A_i$  is simple for all  $i \in [r]$ .

► **Definition 21.** For a tensor  $T \in \mathbb{R}^{n_1 \times n_2 \times \dots \times n_k}$ , the approximate rank of  $T$  is defined as follows:  $\text{rank}_\varepsilon(T) = \min\{\text{rank}(S) \mid \|T - S\|_\infty \leq \varepsilon\}$ . Here  $\|\cdot\|_\infty$  is the entry-wise  $\ell_\infty$ -norm of a tensor.

## 2.4 Quantum Query Complexity

In this section we recall some previous results on quantum query complexity. Here we emphasize the number of qubits used by the algorithms, which will be crucial when simulating them using classical algorithms.

► **Definition 22.** In the FORMULA EVALUATION problem, we are given a formula  $\mathcal{F}$  with  $\{\text{AND}, \text{OR}, \text{NOT}\}$  basis and  $O(1)$  fan-in on  $n$  variables  $x_1, x_2, \dots, x_n$ . In each query, the algorithm gets the value of  $x_i$ , where  $i \in [n]$  is determined by the algorithm. The goal is to evaluate the formula.

► **Theorem 23 ([15]).** *The FORMULA EVALUATION problem can be solved in  $O(n^{1/2+o(1)})$  queries using  $O(\text{polylog}(n))$  qubits, with failure probability at most  $1/3$ .*

► **Remark.** There is an optimal  $O(n^{1/2})$  query algorithm for FORMULA EVALUATION [42]. However, that query algorithm doesn't fit in our applications here for two reasons: (1) the algorithm needs  $O(n)$  qubits, which is too much for classical simulation; (2) the algorithm is not *computationally efficient* and it takes too much time to compute the corresponding unitary transformation.

► **Definition 24.** In the ELEMENT DISTINCTNESS problem, we are given  $n$  elements  $X = (x_1, x_2, \dots, x_n) \in [m]^n$ . In each query, the algorithm gets the value of  $x_i$ , where  $i \in [n]$  is determined by the algorithm. The goal is to decide whether there are two distinct indices  $i \neq j$  such that  $x_i = x_j$ .

► **Theorem 25 ([14]).** *The ELEMENT DISTINCTNESS problem can be solved in  $O(n^{2/3})$  queries using  $O(n^{2/3} \log m)$  qubits, with failure probability at most  $1/3$ .*



## 2.5 Multiparty Quantum Communication Protocols

In this section, we give our definition of multiparty quantum communication protocols.

Let  $\mathcal{X}_1, \mathcal{X}_2, \dots, \mathcal{X}_k$  be finite sets and  $f : \mathcal{X}_1, \mathcal{X}_2, \dots, \mathcal{X}_k \rightarrow \{0, 1\}$  be a function. In a  $k$ -part quantum communication protocols, there are  $k$  players  $P_1, P_2, \dots, P_k$ , together with a Hilbert space  $H = H_1 \otimes H_2 \otimes \dots \otimes H_k \otimes \overline{H}$ . Here  $H_i$  serves as the inner working space for player  $P_i$ , and  $\overline{H}$  is the communication channel between all the players. Each player  $P_i$  receives an input  $x_i \in \mathcal{X}_i$  and the goal is to determine  $f(x_1, x_2, \dots, x_k)$ .

Now we give the formal definition of a  $k$ -party quantum communication protocol.

► **Definition 26.** A  $k$ -part quantum communication protocol  $\mathcal{P} = \mathcal{P}(x_1, x_2, \dots, x_k)$  is a sequence of  $r$  unitary transforms  $\mathcal{P} = (U_1^{p_1}(x_{p_1}), U_2^{p_2}(x_{p_2}), \dots, U_r^{p_r}(x_{p_r}))$ , such that:

- $U_i^{p_i}(x_{p_i})$  is a unitary transform acting on  $H_{p_i} \otimes \overline{H}_i$  where  $\overline{H}_i$  is a subspace spanned by some qubits of  $\overline{H}^{12}$ . That is, it is the action of  $p_i$ -th player  $P_{p_i}$ , who is in charge of the  $i$ -th turn.
- The sequence  $p_1, p_2, \dots, p_r$ , and  $\overline{H}_1, \overline{H}_2, \dots, \overline{H}_r$  are fixed and do not depend on  $x_1, \dots, x_k$ . In other words,  $\overline{H}_i$  corresponds to the qubits in the channel  $\overline{H}$  that player  $P_{p_i}$  will modify during its action in the  $i$ -th turn, and all players take actions in a fixed, predefined order.
- The communication complexity of  $\mathcal{P}$  is defined to be  $C(\mathcal{P}) = \sum_{i=1}^r \log(\dim(\overline{H}_i))$ . The space complexity of  $P_i$  is defined to be  $S_i(\mathcal{P}) = \log(\dim(H_i \otimes \overline{H}))$ .

For a protocol  $\mathcal{P} = (U_1^{p_1}(x_{p_1}), U_2^{p_2}(x_{p_2}), \dots, U_r^{p_r}(x_{p_r}))$ , we say  $\mathcal{P}$  computes  $f$  with error  $\varepsilon$  if we measure the *first* qubit in  $\overline{H}$  on the state  $U_r^{p_r}(x_{p_r}) \cdot U_{r-1}^{p_{r-1}}(x_{p_{r-1}}) \cdot \dots \cdot U_2^{p_2}(x_{p_2}) \cdot U_1^{p_1}(x_{p_1}) \cdot |0\rangle$ , we get  $f(x_1, x_2, \dots, x_k)$  with probability at least  $1 - \varepsilon$ , for all  $x_1 \in \mathcal{X}_1, x_2 \in \mathcal{X}_2, \dots, x_k \in \mathcal{X}_k$ .

► **Remark.** We remark that our definition here is more complicated than the usual definition of quantum communication protocols in the literature (see, e.g., [34]), but nonetheless, it is equivalent to them. We choose to formulate it in such a way because it is easier to describe the classical simulation of quantum communication protocols for low approximate rank decompositions, and the simulation of quantum query algorithms (see below).

### 2.5.1 Simulating Quantum Query Algorithm in Quantum Communication Protocols

Quantum communication protocols can be built upon quantum query algorithms (see, e.g., [20]). Here we give an example to show how to simulate a quantum query algorithm for FORMULA EVALUATION to construct a quantum communication protocol for the communication problem corresponding to Problem 1, under our definition.

In the corresponding  $k$ -party communication problem, there are  $k$  players, and the  $i$ -th player  $P_i$  is given a vector  $u_i \in [r]^d$ . There are  $d$  functions  $f_1, f_2, \dots, f_d$  where each  $f_i$  is from  $[r]^k$  to  $\{0, 1\}$ , and a Boolean formula  $\mathcal{F} : \{0, 1\}^d \rightarrow \{0, 1\}$  of  $O(1)$  fan-in. Set  $v(i) = f_i(u_{1,i}, u_{2,i}, \dots, u_{k,i})$ . Their goal is to compute  $\mathcal{F}(v(1), v(2), \dots, v(d))$ .

Now we show how to construct a quantum communication protocol for the above problem.

► **Example 27.** Assume that the first player runs a quantum query algorithm for the FORMULA EVALUATION problem. For the simulation, we only need to implement the following query gate  $O_v: |i\rangle |b\rangle \rightarrow |i\rangle |b \oplus v(i)\rangle$ , where  $i$  is the index of a variable written in binary form and  $v(i)$  is the corresponding input bit to  $\mathcal{F}$ .

<sup>12</sup> i.e.,  $U_i^{p_i}(x_{p_i})$  does not alter qubits other than those in  $H_{p_i} \otimes \overline{H}_i$ .



We first specify the channel,  $\overline{H}$  is defined as  $\overline{H}_{\text{index}} \otimes \overline{H}_{\text{output}} \otimes \overline{H}_1 \otimes \cdots \otimes \overline{H}_k$ .  $\overline{H}_{\text{index}}$  and  $\overline{H}_{\text{output}}$  together simulate the query gate, and  $\overline{H}_i$  is the place for player  $P_i$  to write her number.

In the beginning, all qubits in  $\overline{H}$  are  $|0\rangle$ . When the first player wants to apply  $O_v$  on some qubits in  $H_1$ , it first swaps the qubits containing  $i$  and  $b$  in  $H_1$  with  $\overline{H}_{\text{index}}$  and  $\overline{H}_{\text{output}}$  in  $\overline{H}$ .

Each player  $P_j$  in turn reads  $i$  in  $\overline{H}_{\text{index}}$  and writes the value of  $u_{j,i}$  to qubits in  $\overline{H}_j$ . Note that each player can write the value of  $u_{j,i}$  to qubits in  $\overline{H}_j$  using a unitary transformation since all qubits in  $\overline{H}_j$  are  $|0\rangle$  at the beginning, by assumption.

Now, given the value of  $i$  and  $u_{1,i}, u_{2,i}, \dots, u_{k,i}$ , the first player maps  $|i\rangle|b\rangle$  to  $|i\rangle|b \oplus v(i)\rangle$  via a unitary transformation. Now the gate  $O_v$  is implemented, but we still have to clean up the garbages in  $\overline{H}_j$ 's, and set them back to  $|0\rangle$ 's. This can be done by applying reverse transforms of all applied unitary transformation, in the reverse order.

The communication complexity of this protocol is  $O(Q \cdot k(\log d + \log r))$ , where  $Q$  is the query complexity of the quantum query algorithm. Also, using the algorithm in Theorem 23, the communication complexity of this protocol is  $O(n^{1/2+o(1)} \cdot k(\log d + \log r))$ .

### 3 Approximate Counting Algorithms from Quantum Communication Protocols

Let  $\mathcal{X}_1, \mathcal{X}_2, \dots, \mathcal{X}_k$  be finite sets and  $f : \mathcal{X}_1, \mathcal{X}_2, \dots, \mathcal{X}_k \rightarrow \{0, 1\}$  be a function. Let  $M_f \in \{0, 1\}^{|\mathcal{X}_1| \times |\mathcal{X}_2| \times \dots \times |\mathcal{X}_k|}$  denote the Boolean tensor whose  $(x_1, x_2, \dots, x_k)$  entry is  $f(x_1, x_2, \dots, x_k)$ . The following connection between 2-party quantum communication complexity and approximate rank is first observed in [21]. This result can be generalized to the  $k$ -party case to get the following theorem. Full details can be found in the full version of this paper.

► **Theorem 28.** *Let  $\mathcal{X}_1, \mathcal{X}_2, \dots, \mathcal{X}_k$  be finite sets and  $f : \mathcal{X}_1, \mathcal{X}_2, \dots, \mathcal{X}_k \rightarrow \{0, 1\}$  be a Boolean function. Suppose there exists a  $k$ -party efficient quantum communication protocol  $\mathcal{P}$ , such that  $\mathcal{P}$  gives the correct answer with probability at least  $1 - \varepsilon$  on every input, then  $\text{rank}_\varepsilon(M_f) \leq 2^{O(C(\mathcal{P}))}$ , or equivalently, there exist simple tensors  $A_1, A_2, \dots, A_{2^{O(C(\mathcal{P}))}}$  such that*

$$\left\| M_f - \sum_{i=1}^{2^{O(C(\mathcal{P}))}} A_i \right\|_\infty \leq \varepsilon.$$

In the full version of this paper, we further show how to use classical deterministic algorithms to simulate quantum communication protocols. Notice that here the time complexity depends on the space complexity of the quantum communication protocol to use.

► **Corollary 29.** *Let  $\mathcal{X}_1, \mathcal{X}_2, \dots, \mathcal{X}_k$  be finite sets and  $f : \mathcal{X}_1, \mathcal{X}_2, \dots, \mathcal{X}_k \rightarrow \{0, 1\}$  be a Boolean function. Suppose there exists a  $k$ -party efficient quantum communication protocol  $\mathcal{P}$ , such that  $\mathcal{P}$  gives the correct answer with probability at least  $1 - \varepsilon$  on every input, and all the unitary transformation used in the  $\mathcal{P}$  can be constructed in polynomial time (with respect to their sizes) by a deterministic classical algorithm. Then there exists  $k$  deterministic classical algorithms  $\mathcal{A}_{\mathcal{X}_1}, \mathcal{A}_{\mathcal{X}_2}, \dots, \mathcal{A}_{\mathcal{X}_k}$  such that  $\mathcal{A}_{\mathcal{X}_i}$  runs in  $2^{O(C(\mathcal{P})+S_i(\mathcal{P}))}$  time, receives  $x_i \in \mathcal{X}_i$  as input and outputs a vector  $\mathcal{A}_{\mathcal{X}_i}(x_i) \in \mathbb{R}^{2^{O(C(\mathcal{P}))}}$ , and for any  $x_1 \in \mathcal{X}_1, x_2 \in \mathcal{X}_2, \dots, x_k \in \mathcal{X}_k$ ,*

$$-\varepsilon \leq \langle \mathcal{A}_{\mathcal{X}_1}(x_1), \mathcal{A}_{\mathcal{X}_2}(x_2), \dots, \mathcal{A}_{\mathcal{X}_k}(x_k) \rangle - f(x_1, x_2, \dots, x_k) \leq \varepsilon.$$

Based on Corollary 29, for any Boolean function  $f : \mathcal{X}_1, \mathcal{X}_2, \dots, \mathcal{X}_k \rightarrow \{0, 1\}$  with an efficient quantum communication protocol, there also exists an efficient approximate counting algorithm for  $f$ .

► **Theorem 30.** *Let  $\mathcal{X}_1, \mathcal{X}_2, \dots, \mathcal{X}_k$  be finite sets and  $f : \mathcal{X}_1, \mathcal{X}_2, \dots, \mathcal{X}_k \rightarrow \{0, 1\}$  be a Boolean function. Suppose there exists a  $k$ -party efficient quantum communication protocol  $\mathcal{P}$ , such that  $\mathcal{P}$  gives the correct answer with probability at least  $1 - \varepsilon$  on every input, and all the unitary transformation used in the  $\mathcal{P}$  can be constructed in polynomial time (with respect to their sizes) by a deterministic classical algorithm. Then there exists a classical deterministic algorithm  $\mathcal{C}$  that receives  $X_1 \subseteq \mathcal{X}_1, X_2 \subseteq \mathcal{X}_2, \dots, X_k \subseteq \mathcal{X}_k$  as input, and outputs a number  $E$  such that*

$$\left| \sum_{x_1 \in X_1, x_2 \in X_2, \dots, x_k \in X_k} f(x_1, x_2, \dots, x_k) - E \right| \leq \varepsilon \cdot \prod_{i=1}^k |X_i|.$$

Furthermore,  $\mathcal{C}$  runs in  $\sum_{i=1}^k |X_i| \cdot 2^{C(\mathcal{P})+S_i(\mathcal{P})}$  time.

**Proof.** For all  $x_i \in \mathcal{X}_i$  we first use  $\mathcal{A}_{\mathcal{X}_i}$  in Corollary 29 to calculate  $\mathcal{A}_{\mathcal{X}_i}(x_i) \in \mathbb{R}^{2^{O(C(\mathcal{P}))}}$ , in  $\sum_{i=1}^k |X_i| \cdot 2^{C(\mathcal{P})+S_i(\mathcal{P})}$  time. Then we directly output

$$\left\langle \sum_{x_1 \in X_1} \mathcal{A}_{\mathcal{X}_1}(x_1), \sum_{x_2 \in X_2} \mathcal{A}_{\mathcal{X}_2}(x_2), \dots, \sum_{x_k \in X_k} \mathcal{A}_{\mathcal{X}_k}(x_k) \right\rangle.$$

The correctness simply follows from the fact that for all  $(x_1, x_2, \dots, x_k) \in \prod_i \mathcal{X}_i$ ,

$$-\varepsilon \leq \langle \mathcal{A}_{\mathcal{X}_1}(x_1), \mathcal{A}_{\mathcal{X}_2}(x_2), \dots, \mathcal{A}_{\mathcal{X}_k}(x_k) \rangle - f(x_1, x_2, \dots, x_k) \leq \varepsilon. \quad \blacktriangleleft$$

► **Remark.** The algorithm described above is actually a sketching algorithm. We may define the sketch for  $X_i$  as  $\text{sk}_i(X_i) = \sum_{x_i \in X_i} \mathcal{A}_{\mathcal{X}_i}(x_i) \in \mathbb{R}^{2^{O(C(\mathcal{P}))}}$  and the number  $E$  can be computed from these  $\text{sk}_i$ 's. This sketching algorithm satisfies a nice additive property, i.e., the sketch of  $A \sqcup B$  (union as a multi-set) is simply  $\text{sk}_i(A) + \text{sk}_i(B)$ .

Now we give approximate counting algorithms for concrete problems, using Theorem 30.

### 3.1 Counting the $k$ -Tuples of Orthogonal Vectors

The goal of this section is to prove the following theorem.

**Reminder of Theorem 4.** *For any integers  $k, d$  and any constant  $\varepsilon > 0$ ,  $\#k\text{-OV}_{n,d}$  can be approximated deterministically with additive error  $\varepsilon \cdot n^k$  in  $n \cdot 2^{O(k\sqrt{d})}$  time. In particular, it runs in  $n^{1+o(1)}$  time  $k$  is a constant and  $d = o(\log^2 n)$ .*

We first consider quantum communication protocols for the following function  $f$ .

► **Definition 31.** Let  $\mathcal{X}_1 = \mathcal{X}_2 = \dots = \mathcal{X}_k = \{0, 1\}^d$  and

$$f(x_1, x_2, \dots, x_k) = \begin{cases} 1 & \text{if } \langle x_1, x_2, \dots, x_k \rangle = 0 \\ 0 & \text{otherwise} \end{cases}.$$

The corresponding communication problem can be solved using the quantum communication protocol in [1] with communication complexity  $O(k\sqrt{d})$  and space complexity  $O(\text{polylog}(d))$ , with constant failure probability. If we use the algorithm in Theorem 30, together with the efficient quantum communication protocol mentioned above, we can then deterministically count the number of  $k$ -tuples of orthogonal vectors, in time  $n \cdot 2^{O(k\sqrt{d})}$  time, with an additive  $\varepsilon \cdot n^k$  error.

### 3.2 Counting the Pairs of Orthogonal Sparse Vectors

The goal of this section is to prove the following theorem.

**Reminder of Theorem 5.** For integers  $n, m, d$  and any constant  $\varepsilon > 0$ ,  $\#\text{Sparse-OV}_{n,m,d}$  can be approximated deterministically with additive error  $\varepsilon \cdot n^2$  in

$$n \cdot 2^{O(d^{2/3} \log(m))}$$

time. In particular, when  $m = \text{poly}(d)$  and  $d = o\left(\left(\frac{\log n}{\log \log n}\right)^{1.5}\right)$ , it runs in  $n^{1+o(1)}$  time.

Again we consider quantum communication protocols for the following function  $f$ .

► **Definition 32.** Let  $\mathcal{X} = \mathcal{Y} = \{0, 1\}_{\leq d}^m$  and

$$f(x, y) = \begin{cases} 1 & \text{if } \langle x, y \rangle = 0 \\ 0 & \text{otherwise} \end{cases}.$$

The corresponding communication problem can be solved with communication complexity  $O(d^{2/3} \log m)$ , by simulating the quantum query algorithm in Theorem 25 for ELEMENT DISTINCTNESS. To see the connection, let  $S = \{i \mid x_i = 1\}$  and  $T = \{i \mid y_i = 1\}$ . We will have  $f(x, y) = 1$  if and only if all elements in  $S \sqcup T$  (union as a multi-set) are distinct. Now, using the algorithm in Theorem 30, together with the efficient quantum communication protocol mentioned above, we can deterministically count the number of orthogonal pairs in  $S$  and  $T$ , in  $n \cdot 2^{O(d^{2/3} \log(m))}$  time, with an additive  $\varepsilon \cdot n^k$  error.

### 3.3 Counting Solutions to Formula $\circ$ SYM Circuits

The goal of this section is to solve the following problem.

**Reminder of Problem 1.** Given  $k$  sets of  $n$  vectors  $S_1, S_2, \dots, S_k \subseteq \{0, \dots, r\}^d$  and  $d$  functions  $f_1, f_2, \dots, f_d$  where each  $f_i$  is from  $\{0, \dots, r\}^k$  to  $\{0, 1\}$ , and a Boolean formula  $\mathcal{F} : \{0, 1\}^d \rightarrow \{0, 1\}$  of  $O(1)$  fan-in. Count the number of  $k$ -tuples  $u_1 \in S_1, u_2 \in S_2, \dots, u_k \in S_k$  such that

$$\mathcal{F}(f_1(u_{1,1}, u_{2,1}, \dots, u_{k,1}), f_2(u_{1,2}, u_{2,2}, \dots, u_{k,2}), \dots, f_d(u_{1,d}, u_{2,d}, \dots, u_{k,d})) = 1.$$

**Reminder of Theorem 7.** For any constant  $\varepsilon > 0$ , the above problem can be solved deterministically in  $n \cdot 2^{O(d^{1/2+o(1)} \cdot k(\log d + \log r))}$  time, within  $\varepsilon \cdot n^k$  additive error.

The corresponding  $k$ -party communication problem can be solved by a quantum communication protocol with communication complexity  $O(d^{1/2+o(1)} \cdot k(\log d + \log r))$ , by simulating the quantum query algorithm for Formula-Evaluation in Theorem 23. For details see Example 27. By our framework, this implies an approximate counting algorithm to the problem mentioned above in time  $n \cdot 2^{O(d^{1/2+o(1)} \cdot k(\log d + \log r))}$ , with an additive  $\varepsilon \cdot n^k$  error.

Here we mention one application to the approximate counting algorithm above.

**Reminder of Theorem 6.** For any constant  $\varepsilon > 0$ , the number of solutions to a Formula  $\circ$  SYM circuit of size  $m$  can be approximated deterministically within  $\varepsilon \cdot 2^n$  additive error in  $2^{O(n^{1/2} m^{1/4+o(1)} \sqrt{\log n + \log m})}$  time. In particular, when  $m = n^{2-\delta}$  for some  $\delta > 0$ , the running time is  $2^{o(n)}$ .

**Proof of Theorem 6.** Consider a Formula  $\circ$  SYM circuit  $\mathcal{C} : \{0, 1\}^n \rightarrow \{0, 1\}$  with  $m$  symmetric gates  $X_1, X_2, \dots, X_m$  and a Boolean formula  $\mathcal{F}$  of  $O(1)$  fan-in. Here we slightly abuse of notation by regarding  $X_i$  as a function that maps the number of inputs bits with value one to an output in  $\{0, 1\}$ . We can approximately count the number of solutions to  $\mathcal{C}$  as follows.

We split the  $n$  inputs bits into  $s$  groups, each with  $n/s$  input bits. Then for each group, we enumerate all the  $2^{n/s}$  possible assignments to the  $n/s$  input bits. We create a vector in  $\{0, \dots, n/s\}^m$  for each possible assignment, where the  $i$ -th entry is simply the number of ones in the assignment which is an input bit to the  $i$ -th symmetric gate  $X_i$ . Now, the number of solutions to the circuit  $\mathcal{C}$ , is simply the same as Problem 1, by setting

$$f_i(u_{1,i}, u_{2,i}, \dots, u_{k,i}) = X_i(u_{1,i} + u_{2,i} + \dots + u_{k,i}).$$

The total time complexity would be  $2^{n/s} \cdot 2^{O(m^{1/2+o(1)} \cdot s(\log m + \log(n/s)))}$ , with an additive  $\varepsilon \cdot 2^n$  error. Setting  $s = \frac{n^{1/2}}{m^{1/4+o(1)} \sqrt{\log n + \log m}}$ , the final time complexity would be

$$2^{O(n^{1/2} m^{1/4+o(1)} \sqrt{\log n + \log m})}. \quad \blacktriangleleft$$

## 4 Algorithms from Arthur-Merlin Communication Protocols

In this section, we prove our algorithmic applications of  $\text{AM}^{\text{cc}}$  protocols. We first show faster  $\text{AM}^{\text{cc}}$  protocols for  $F$  imply faster  $F$ -Satisfying-Pair algorithms.

**Reminder of Theorem 9.** *Let  $F : \mathcal{X} \times \mathcal{Y} \rightarrow \{0, 1, \perp\}$  be a partial function. Suppose there is a computationally efficient  $\text{AM}^{\text{cc}}$  protocol for  $F$  with communication complexity  $T$  and error  $\varepsilon$ . Then for  $n$  such that  $2^T \leq (\sqrt{\varepsilon}n)^{0.1}$ , there is an  $O(\varepsilon n^2 \cdot \text{polylog}(n) + n \cdot 2^T)$  time randomized algorithm for  $F$ -Satisfying-Pair $_n$ .*

**Proof.** We first assume  $n < \frac{1}{10\sqrt{\varepsilon}}$ . After drawing a random challenge, for each element  $x \in \mathcal{X}$  and  $y \in \mathcal{Y}$  we construct a Boolean vector  $\mathcal{A}_x(x)$  and  $\mathcal{A}_y(y)$  of length  $2^T$ , where each the  $i$ -th entry indicates whether Alice (Bob) accepts when receiving the proof  $i$  from Merlin. Here we regard  $i$  as a Boolean string of length  $T$  via a natural bijection between  $[2^T]$  and  $\{0, 1\}^T$ .

According to the guarantee of an  $\text{AM}^{\text{cc}}$  protocol, for each  $x \in \mathcal{X}$  and  $y \in \mathcal{Y}$ , when  $F(x, y) = 1$ , with probability at least  $1 - \varepsilon$  over the random challenge, we have  $\langle \mathcal{A}_x(x), \mathcal{A}_y(y) \rangle > 0$ , and when  $F(a, b) = 0$  we have  $\langle \mathcal{A}_x(x), \mathcal{A}_y(y) \rangle > 0$  with probability at most  $\varepsilon$  over the random challenge.

By a union bound on all pairs of elements in  $A$  and  $B$ , we have with probability at least 0.99, for all  $a \in A$  and  $b \in B$ ,  $\langle \mathcal{A}_A(a), \mathcal{A}_B(b) \rangle > 0$  if and only if  $F(a, b) = 1$ . Consequently, with probability at least 0.99,

$$\left\langle \sum_{a \in A} \mathcal{A}_A(a), \sum_{b \in B} \mathcal{A}_B(b) \right\rangle > 0$$

if and only if there exist  $a \in A$  and  $b \in B$  such that  $F(a, b) = 1$ .

For general  $n = |A| = |B|$ , we first split  $A$  and  $B$  into  $O(\sqrt{\varepsilon}n)$  groups, each with at most  $\frac{1}{10\sqrt{\varepsilon}}$  elements. I.e., we assume  $A = \bigcup_{i=1}^g A_i$  and  $B = \bigcup_{i=1}^g B_i$  such that  $g = O(\sqrt{\varepsilon}n)$  and  $|A_i|, |B_i| \leq \frac{1}{10\sqrt{\varepsilon}}$ . For each  $i, j \in [g]$ , we use the algorithm mentioned above to calculate two vectors  $\sum_{a \in A_i} \mathcal{A}_A(a)$  and  $\sum_{b \in B_j} \mathcal{A}_B(b)$ . We write  $\mathcal{M}_A \in \mathbb{R}^{2^T \times g}$  to denote the matrix

$$\left[ \sum_{a \in A_1} \mathcal{A}_A(a), \sum_{a \in A_2} \mathcal{A}_A(a), \dots, \sum_{a \in A_g} \mathcal{A}_A(a) \right]$$

and  $\mathcal{M}_B \in \mathbb{R}^{2^T \times g}$  to denote the matrix

$$\left[ \sum_{b \in B_1} \mathcal{A}_B(b), \sum_{b \in B_2} \mathcal{A}_B(b), \dots, \sum_{b \in B_g} \mathcal{A}_B(b) \right].$$

Since  $2^T \leq (\sqrt{\varepsilon}n)^{0.1} \leq O(g^{0.1})$ , we can use the rectangular matrix multiplication algorithm in Theorem 16 to calculate  $\mathcal{M}_A^T \mathcal{M}_B$  in  $O(g^2 \cdot \text{polylog}(g)) = O(\varepsilon n^2 \text{polylog}(n))$  time. We repeat this procedure for  $O(\log n)$  times. For any  $i, j \in [g]$ , by standard concentration bounds, with probability at least  $1 - \text{poly}(n)$ , there exist  $a \in A_i$  and  $b \in B_j$  such that  $F(a, b) = 1$  if and only if the majority of the  $O(\log n)$  repetitions satisfies  $(\mathcal{M}_A^T \mathcal{M}_B)_{i,j} > 0$ . Applying union bound again over all  $i, j \in [g]$ , we can now solve  $F$ -Satisfying-Pair $_n$  by checking whether there exist  $i$  and  $j$  such that the majority of the  $O(\log n)$  repetitions satisfies  $(\mathcal{M}_A^T \mathcal{M}_B)_{i,j} > 0$ . The overall algorithm runs in  $O(\varepsilon n^2 \cdot \text{polylog}(n))$  time and succeeds with high probability, as stated.  $\blacktriangleleft$

#### 4.1 A New Algorithm for Approximate Max-IP

The first application of Theorem 9 is to use the Goldwasser-Sispe AM protocol [28] for approximating set size to obtain a new algorithm for approximating Max-IP.

We first need the following adaption of [28], which has a better dependence on  $\varepsilon$ .

**Reminder of Lemma 11.** *There is an  $\text{AM}^{\text{cc}}$  protocol for  $\text{Gap-Inner-Product}_d$  with error  $\varepsilon$  and communication complexity  $\log \binom{d}{\leq O(\log \varepsilon^{-1})}$ .*

**Proof.** Recall that  $x, y \in \{0, 1\}^d$  are the inputs hold by Alice and Bob respectively.

Let  $X = \{i \mid x_i = 1\}$  and  $Y = \{i \mid y_i = 1\}$ . The problem is equivalent to determine whether  $|X \cap Y| \geq 2\tau$  or  $|X \cap Y| \leq \tau$ . Here we give an  $\text{AM}^{\text{cc}}$  communication protocol with error  $\varepsilon$  and communication complexity  $\log \binom{d}{\leq O(\log \varepsilon^{-1})}$ .

In the communication protocol, Alice and Bob first generate i.i.d. random variables  $p_i \sim \text{Pois}(k/\tau)$  for each  $i \in [d]$ , for a parameter  $k = \Theta(\log(1/\varepsilon))$  to be determined later. When  $|X \cap Y| \geq 2\tau$ , Merlin finds an arbitrary set  $S \subseteq X \cap Y$  of size  $O(k)$  such that  $\sum_{i \in S} p_i \geq 1.6k$ , and then sends it to Alice and Bob. Upon receiving  $S$ , Alice (Bob) decides to accept or reject by checking whether  $S \subseteq X$  ( $S \subseteq Y$ ) and  $\sum_{i \in S} p_i \geq 1.6k$ . The communication complexity of this protocol is upper bounded by  $\log \binom{d}{\leq O(\log \varepsilon^{-1})}$  since  $|S| \leq 1.6k = O(\log(1/\varepsilon))$ .

Now we prove the correctness by considering the following two cases.

**Case 1:**  $|X \cap Y| \geq 2\tau$ . For this case, we have  $\sum_{i \in X \cap Y} p_i \sim \text{Pois}(|X \cap Y| \cdot k/\tau) \succeq \text{Pois}(2k)$ .

Thus by Lemma 18, with probability at least  $1 - e^{-\Omega(k)}$ ,  $\sum_{i \in X \cap Y} p_i \geq 1.6k$ . Since for each  $p_i > 0$  we must have  $p_i \geq 1$ , with probability at least  $1 - e^{-\Omega(k)}$ , there exists a set  $S \subseteq X \cap Y$  of size  $O(k)$  such that  $\sum_{i \in S} p_i \geq 1.6k$ .

**Case 2:**  $|X \cap Y| \leq \tau$ . For this case, we have  $\sum_{i \in X \cap Y} p_i \sim \text{Pois}(|X \cap Y| \cdot k/\tau) \preceq \text{Pois}(k)$ .

Thus by Lemma 18, with probability at least  $1 - e^{-\Omega(k)}$ ,  $\sum_{i \in X \cap Y} p_i \leq 1.2k$ . When both Alice and Bob accept, it must be the case that  $S \subseteq X \cap Y$  and  $\sum_{i \in S} p_i \geq 1.6k$ . However when  $|X \cap Y| \leq \tau$ , with probability at least  $1 - e^{-\Omega(k)}$ ,  $\sum_{i \in X \cap Y} p_i \leq 1.2k$ . Thus there is no  $S$  such that both Alice and Bob accept, with probability at least  $1 - e^{-\Omega(k)}$ .

The lemma follows by setting  $k$  to be a large enough multiple of  $\log(1/\varepsilon)$ .  $\blacktriangleleft$

By Theorem 9 and the above lemma, Corollary 12 follows from a binary search over  $\tau$ .

**Reminder of Corollary 12.** *There is an algorithm for computing a 2-approximation to  $\text{Max-IP}_{n,c \log n}$ , which runs in  $n^{2-1/O(\log c)}$  time.*

## 4.2 Consequence of Fast $\text{AM}^{\text{cc}}$ Protocols for LCS and Edit-Distance

Next we discuss the consequences of LCS and Edit-Distance having efficient  $\text{AM}^{\text{cc}}$  protocols. We first introduce some classical notations about the communication complexity classes (see [17, 30]). We say a function family  $F = \{F_d : \{0, 1\}^d \times \{0, 1\}^d \rightarrow \{0, 1, \perp\}\}_{d \in \mathbb{N}}$  is in  $\text{AM}^{\text{cc}}$  if  $\text{AM}^{\text{cc}}(F_d) = \text{polylog}(d)$  (we use  $\text{AM}^{\text{cc}}(F_d)$  to denote the  $\text{AM}^{\text{cc}}$  communication complexity for  $F_d$  with error  $1/3$ ).

We also say  $F$  is  $\text{AM}_{\text{eff}}^{\text{cc}}$  if for all  $d \in \mathbb{N}$ ,  $F_d$  admits a computationally efficient  $\text{AM}^{\text{cc}}$  protocol with error  $1/3$  and complexity  $\text{polylog}(d)$ .

Now we prove the consequence of a function family  $F \in \text{AM}_{\text{eff}}^{\text{cc}}$ .

► **Corollary 33** (Consequence of  $F \in \text{AM}_{\text{eff}}^{\text{cc}}$ ). *Let  $F = \{F_d : \{0, 1\}^d \times \{0, 1\}^d \rightarrow \{0, 1, \perp\}\}_{d \in \mathbb{N}}$  be a partial function family. If  $F \in \text{AM}_{\text{eff}}^{\text{cc}}$ , then there is an  $n^2/2^{\log^{1-\delta} n}$  time algorithm for  $F_{\text{polylog}(n)}$ -Satisfying-Pair $_n$ , for any constant  $\delta > 0$ .*

**Proof.** By standard repetition arguments, there exists an  $\text{AM}^{\text{cc}}$  communication protocol with communication complexity  $\text{polylog}(d) \log(1/\varepsilon)$  and failure probability  $1 - \varepsilon$ . In order to invoke Theorem 9 we need to make sure

$$2^{\text{polylog}(d) \log(1/\varepsilon)} = 2^{\text{polyloglog}(n) \log(1/\varepsilon)} < n^{0.1},$$

and thus we can set  $\varepsilon = 2^{-\log^{1-\delta/2} n}$ . For this choice of  $\varepsilon$  we will then get an  $n^2/2^{\log^{1-\delta/2} n}$  time algorithm for  $F_{\text{polylog}(n)}$ -Satisfying-Pair $_n$ , which completes the proof. ◀

Recall that in  $\text{LCS}_d^{\text{cc}}$  ( $\text{Edit-Dist}_d^{\text{cc}}$ ), Alice and Bob hold strings  $x, y \in \{0, 1\}^d$  respectively, and are given an integer  $\tau$ . Their goal is to decide whether  $\text{LCS}(x, y)$  is at least  $\tau$  ( $\text{Edit-Distance}(x, y)$  is at least  $\tau$ ). Now we are ready to prove Theorem 13.

**Reminder of Theorem 13.** *If  $\text{LCS}_d^{\text{cc}}$  admits computationally efficient  $\text{AM}^{\text{cc}}$  protocols with complexity  $\text{polylog}(d)$ , then Formula-SAT of polynomial-size formulas admits an  $2^{n-n^{1-\delta}}$  time algorithm for any constant  $\delta > 0$ . The same holds for  $\text{Edit-Dist}^{\text{cc}}$  in place of  $\text{LCS}^{\text{cc}}$ .*

We will only discuss  $\text{LCS}^{\text{cc}}$  here, the proof for  $\text{Edit-Dist}^{\text{cc}}$  follows exactly the same. We first introduce the reduction from [3] (see also [2]).

► **Theorem 34** (Implicit in [3]). *For a given formula  $\mathcal{F}$  with  $n$  input variables and size  $s$ , let  $a \in \{0, 1\}^{n/2}$  be an assignment to first  $n/2$  variables in  $\mathcal{F}$  and  $b \in \{0, 1\}^{n/2}$  be an assignment to last  $n/2$  variables in  $\mathcal{F}$ . There exists an algorithm  $\mathcal{A}$  which outputs  $G(a) \in \{0, 1\}^{\text{poly}(s)}$  and  $\overline{G}(b) \in \{0, 1\}^{\text{poly}(s)}$  such that for a fixed integer  $Y$  ( $Y$  depends on  $\mathcal{F}$ ),*

- $\text{LCS}(G(a), \overline{G}(b)) = Y$  if  $a \odot b$  is a satisfying assignment to  $\mathcal{F}$ ;
- $\text{LCS}(G(a), \overline{G}(b)) \leq Y - 1$  if  $a \odot b$  is not a satisfying assignment to  $\mathcal{F}$ .

**Proof of Theorem 13.** For a given formula  $\mathcal{F}$  of size  $s = \text{poly}(n)$ , we first enumerate all  $2^{n/2}$  possible assignments to first  $n/2$  variables in  $\mathcal{F}$  and all possible assignments to last  $n/2$  variables in  $\mathcal{F}$ . For each  $a \in \{0, 1\}^{n/2}$  corresponding to an assignment to first  $n/2$  variables in  $\mathcal{F}$  and  $b \in \{0, 1\}^{n/2}$  corresponding to an assignment to last  $n/2$  variables in  $\mathcal{F}$ , we calculate  $G(a)$  and  $\overline{G}(b)$  using Theorem 34. Note that all  $G(a)$ 's and  $\overline{G}(b)$ 's have length  $\text{poly}(s) = \text{poly}(n)$ .

Now suppose  $\text{LCS}^{\text{cc}} \in \text{AM}_{\text{eff}}^{\text{cc}}$  for  $\tau = Y$ . Applying Corollary 33 with all possible  $G(a)$ 's and  $\overline{G}(b)$ 's, we can solve Formula-SAT in  $2^{n-n^{1-\delta}}$  time for any constant  $\delta > 0$ . ◀



## Open Problems and Future Directions

Here we list a few interesting open problems stemming from this work.

- In this work, we applied  $\text{BQP}^{\text{cc}}$  and  $\text{AM}^{\text{cc}}$  protocols for the algorithmic purpose. Can we find algorithmic applications of other communication protocols?
- Or less ambitiously, can we find more interesting algorithmic applications with other known  $\text{BQP}^{\text{cc}}$  or  $\text{AM}^{\text{cc}}$  protocols? (this could even be a motivation to find *new*  $\text{BQP}^{\text{cc}}$  or  $\text{AM}^{\text{cc}}$  protocols!)
- Our additive approximation algorithm for  $\#\text{OV}$  runs in near-linear time when  $d = o(\log^2 n)$ . Is it possible to design a near-linear time algorithm for  $d = n^{o(1)}$  dimensions? Note that by a simple Chernoff bound, there is a deterministic  $n^{1+o(1)}$  time algorithm with  $n^{1+o(1)}$  advice for additive approximations to  $\#\text{OV}$ . So there is a hope to construct such an algorithm.
- Our results show that under the hypothesis of [2],  $\text{LCS}^{\text{cc}}$  and  $\text{Edit-Dist}^{\text{cc}}$  do not admit computationally efficient  $\text{AM}^{\text{cc}}$  or  $\text{PH}^{\text{cc}}$  protocols. Can one prove that *unconditionally*?
- Is it possible to connect these algorithms from  $\text{AM}^{\text{cc}}$  or  $\text{PH}^{\text{cc}}$  protocols to R. Williams' algorithmic approach to circuit lower bounds [49, 51, 38]? In particular, can one show *unconditionally* that, there is a function  $f$  in  $\text{NEXP}$  (or even  $\text{NTIME}[2^{\text{polylog}(n)}]$ ), which doesn't admit  $\text{polylog}(n)$  complexity  $\text{AM}^{\text{cc}}$  or  $\text{PH}^{\text{cc}}$  protocols?

---

## References

- 1 Scott Aaronson and Andris Ambainis. Quantum Search of Spatial Regions. *Theory of Computing*, 1(1):47–79, 2005.
- 2 Amir Abboud and Karl Bringmann. Tighter Connections Between Formula-SAT and Shaving Logs. In *45th International Colloquium on Automata, Languages, and Programming, ICALP 2018, July 9-13, 2018, Prague, Czech Republic*, pages 8:1–8:18, 2018.
- 3 Amir Abboud, Thomas Dueholm Hansen, Virginia Vassilevska Williams, and Ryan Williams. Simulating branching programs with edit distance and friends: or: a polylog shaved is a lower bound made. In *Proc. of the 48th STOC*, pages 375–388, 2016.
- 4 Amir Abboud and Aviad Rubinfeld. Fast and Deterministic Constant Factor Approximation Algorithms for LCS Imply New Circuit Lower Bounds. In *9th Innovations in Theoretical Computer Science Conference, ITCS 2018, January 11-14, 2018, Cambridge, MA, USA*, pages 35:1–35:14, 2018.
- 5 Amir Abboud, Aviad Rubinfeld, and R. Ryan Williams. Distributed PCP Theorems for Hardness of Approximation in P. In *58th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2017, Berkeley, CA, USA, October 15-17, 2017*, pages 25–36, 2017.
- 6 Amir Abboud, Richard Ryan Williams, and Huacheng Yu. More Applications of the Polynomial Method to Algorithm Design. In *Proceedings of the Twenty-Sixth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2015, San Diego, CA, USA, January 4-6, 2015*, pages 218–230, 2015.
- 7 Josh Alman. An Illuminating Algorithm for the Light Bulb Problem. In *SOSA*, 2019.
- 8 Josh Alman, Timothy M. Chan, and R. Ryan Williams. Polynomial Representations of Threshold Functions and Algorithmic Applications. In *IEEE 57th Annual Symposium on Foundations of Computer Science, FOCS 2016, 9-11 October 2016, Hyatt Regency, New Brunswick, New Jersey, USA*, pages 467–476, 2016.
- 9 Josh Alman and R. Ryan Williams. Probabilistic rank and matrix rigidity. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2017, Montreal, QC, Canada, June 19-23, 2017*, pages 641–652, 2017.

- 10 Josh Alman and Ryan Williams. Probabilistic Polynomials and Hamming Nearest Neighbors. In *IEEE 56th Annual Symposium on Foundations of Computer Science, FOCS 2015, Berkeley, CA, USA, 17-20 October, 2015*, pages 136–150, 2015.
- 11 Noga Alon. Perturbed identity matrices have high rank: Proof and applications. *Combinatorics, Probability and Computing*, 18(1-2):3–15, 2009.
- 12 Noga Alon, Troy Lee, and Adi Shraibman. The cover number of a matrix and its algorithmic applications. In *LIPICs-Leibniz International Proceedings in Informatics*, volume 28. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2014.
- 13 Noga Alon, Troy Lee, Adi Shraibman, and Santosh Vempala. The approximate rank of a matrix and its algorithmic applications: approximate rank. In *Proceedings of the forty-fifth annual ACM symposium on Theory of computing*, pages 675–684. ACM, 2013.
- 14 Andris Ambainis. Quantum walk algorithm for element distinctness. *SIAM Journal on Computing*, 37(1):210–239, 2007.
- 15 Andris Ambainis, Andrew M Childs, Ben W Reichardt, Robert Špalek, and Shengyu Zhang. Any AND-OR formula of size  $N$  can be evaluated in time  $N^{1/2+o(1)}$  on a quantum computer. *SIAM Journal on Computing*, 39(6):2513–2530, 2010.
- 16 Andris Ambainis, Leonard J. Schulman, Amnon Ta-Shma, Umesh V. Vazirani, and Avi Wigderson. The Quantum Communication Complexity of Sampling. *SIAM J. Comput.*, 32(6):1570–1585, 2003.
- 17 László Babai, Peter Frankl, and Janos Simon. Complexity classes in communication complexity theory (preliminary version). In *27th Annual Symposium on Foundations of Computer Science, Toronto, Canada, 27-29 October 1986*, pages 337–347, 1986.
- 18 Boaz Barak, Zeev Dvir, Amir Yehudayoff, and Avi Wigderson. Rank bounds for design matrices with applications to combinatorial geometry and locally correctable codes. In *Proceedings of the forty-third annual ACM symposium on Theory of computing*, pages 519–528. ACM, 2011.
- 19 Harry Buhrman, Richard Cleve, and Avi Wigderson. Quantum vs. Classical Communication and Computation. In *Proc. of the Thirtieth Annual ACM Symposium on the Theory of Computing*, pages 63–68, 1998.
- 20 Harry Buhrman, Richard Cleve, and Avi Wigderson. Quantum vs. classical communication and computation. In *Proceedings of the thirtieth annual ACM symposium on Theory of computing*, pages 63–68. ACM, 1998.
- 21 Harry Buhrman and Ronald de Wolf. Communication complexity lower bounds by polynomials. In *Computational Complexity, 16th Annual IEEE Conference on, 2001.*, pages 120–130. IEEE, 2001.
- 22 Timothy M. Chan and Ryan Williams. Deterministic APSP, Orthogonal Vectors, and More: Quickly Derandomizing Razborov-Smolensky. In *Proceedings of the Twenty-Seventh Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2016, Arlington, VA, USA, January 10-12, 2016*, pages 1246–1255, 2016.
- 23 Lijie Chen. On The Hardness of Approximate and Exact (Bichromatic) Maximum Inner Product. In *33rd Computational Complexity Conference, CCC 2018, June 22-24, 2018, San Diego, CA, USA*, pages 14:1–14:45, 2018.
- 24 Lijie Chen, Shafi Goldwasser, Kaifeng Lyu, Guy Rothblum, and Aviad Rubinfeld. Fine-grained Complexity Meets IP = PSPACE. In *SODA*, 2019.
- 25 Lijie Chen and Ryan Williams. An Equivalence Class for Orthogonal Vectors. In *SODA*, 2019.
- 26 Don Coppersmith. Rapid multiplication of rectangular matrices. *SIAM Journal on Computing*, 11(3):467–471, 1982.



- 27 Francois 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*, pages 1029–1046. SIAM, 2018.
- 28 Shafi Goldwasser and Michael Sipser. Private Coins versus Public Coins in Interactive Proof Systems. *Advances in Computing Research*, 5:73–90, 1989.
- 29 Mika Göös, Toniann Pitassi, and Thomas Watson. Zero-Information Protocols and Unambiguity in Arthur-Merlin Communication. *Algorithmica*, 76(3):684–719, 2016.
- 30 Mika Göös, Toniann Pitassi, and Thomas Watson. The Landscape of Communication Complexity Classes. *Computational Complexity*, 27(2):245–304, 2018.
- 31 Parikshit Gopalan, Ryan O’Donnell, Yi Wu, and David Zuckerman. Fooling Functions of Halfspaces under Product Distributions. In *Proceedings of the 25th Annual IEEE Conference on Computational Complexity, CCC 2010, Cambridge, Massachusetts, USA, June 9-12, 2010*, pages 223–234, 2010.
- 32 Prahladh Harsha, Adam R. Klivans, and Raghu Meka. An invariance principle for polytopes. *J. ACM*, 59(6):29:1–29:25, 2012.
- 33 Karthik C. S., Bundit Laekhanukit, and Pasin Manurangsi. On the parameterized complexity of approximating dominating set. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2018, Los Angeles, CA, USA, June 25-29, 2018*, pages 1283–1296, 2018.
- 34 Ilan Kremer. *Quantum communication*. Citeseer, 1995.
- 35 Troy Lee and Adi Shraibman. Lower Bounds in Communication Complexity. *Foundations and Trends in Theoretical Computer Science*, 3(4):263–398, 2009.
- 36 Daniel Lokshtanov, Ramamohan Paturi, Suguru Tamaki, R. Ryan Williams, and Huacheng Yu. Beating Brute Force for Systems of Polynomial Equations over Finite Fields. In *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2017, Barcelona, Spain, Hotel Porta Fira, January 16-19*, pages 2190–2202, 2017.
- 37 Kurt Mehlhorn and Erik Meineche Schmidt. Las Vegas Is better than Determinism in VLSI and Distributed Computing (Extended Abstract). In *Proceedings of the 14th Annual ACM Symposium on Theory of Computing, May 5-7, 1982, San Francisco, California, USA*, pages 330–337, 1982.
- 38 Cody Murray and R. Ryan Williams. Circuit lower bounds for nondeterministic quasipolytime: an easy witness lemma for NP and NQP. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2018, Los Angeles, CA, USA, June 25-29, 2018*, pages 890–901, 2018.
- 39 Ryan O’Donnell, Rocco A. Servedio, and Li-Yang Tan. Fooling Polytopes. *CoRR*, abs/1808.04035, 2018.
- 40 Ramamohan Paturi and Janos Simon. Probabilistic Communication Complexity. *J. Comput. Syst. Sci.*, 33(1):106–123, 1986.
- 41 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.
- 42 Ben Reichardt. Reflections for quantum query algorithms. In *Proceedings of the Twenty-Second Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2011, San Francisco, California, USA, January 23-25, 2011*, pages 560–569, 2011.
- 43 Karthik C. S. and Pasin Manurangsi. On Closest Pair in Euclidean Metric: Monochromatic is as Hard as Bichromatic. In *ITCS*, 2019.
- 44 Rocco A. Servedio and Li-Yang Tan. Fooling Intersections of Low-Weight Halfspaces. In *58th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2017, Berkeley, CA, USA, October 15-17, 2017*, pages 824–835, 2017.

- 45 Roman Smolensky. Algebraic Methods in the Theory of Lower Bounds for Boolean Circuit Complexity. In *Proceedings of the 19th Annual ACM Symposium on Theory of Computing, 1987, New York, New York, USA*, pages 77–82, 1987.
- 46 Zhao Song, David P Woodruff, and Peilin Zhong. Relative Error Tensor Low Rank Approximation. In *SODA*, 2019.
- 47 Avishay Tal. #SAT algorithms from shrinkage. *Electronic Colloquium on Computational Complexity (ECCC)*, 22:114, 2015.
- 48 Richard Ryan Williams. The Polynomial Method in Circuit Complexity Applied to Algorithm Design (Invited Talk). In *34th International Conference on Foundation of Software Technology and Theoretical Computer Science, FSTTCS 2014, December 15-17, 2014, New Delhi, India*, pages 47–60, 2014.
- 49 Ryan Williams. Improving exhaustive search implies superpolynomial lower bounds. *SIAM Journal on Computing*, 42(3):1218–1244, 2013.
- 50 Ryan Williams. Faster all-pairs shortest paths via circuit complexity. In *Proceedings of the forty-sixth annual ACM symposium on Theory of computing*, pages 664–673. ACM, 2014.
- 51 Ryan Williams. Non-Uniform ACC circuit lower bounds. *Journal of the ACM (JACM)*, 61(1):2, 2014.
- 52 Huacheng Yu. Optimal Succinct Rank Data Structure via Approximate Nonnegative Tensor Decomposition. *arXiv preprint*, 2018. [arXiv:1811.02078](https://arxiv.org/abs/1811.02078).

# Capturing Complementarity in Set Functions by Going Beyond Submodularity/Subadditivity

Wei Chen<sup>1</sup>

Microsoft Research, Beijing, China  
weic@microsoft.com

Shang-Hua Teng<sup>2</sup>

USC, Los Angeles, CA, USA  
shanghua@usc.edu

Hanrui Zhang<sup>3</sup>

Duke University, Durham, NC, USA  
hrzhang@cs.duke.edu

---

## Abstract

---

We introduce two new “degree of complementarity” measures: *supermodular width* and *superadditive width*. Both are formulated based on natural witnesses of complementarity. We show that both measures are robust by proving that they, respectively, characterize the gap of monotone set functions from being submodular and subadditive. Thus, they define two new hierarchies over monotone set functions, which we will refer to as Supermodular Width (SMW) hierarchy and Superadditive Width (SAW) hierarchy, with foundations – i.e. level 0 of the hierarchies – resting exactly on submodular and subadditive functions, respectively.

We present a comprehensive comparative analysis of the SMW hierarchy and the Supermodular Degree (SD) hierarchy, defined by Feige and Izsak. We prove that the SMW hierarchy is strictly more expressive than the SD hierarchy: Every monotone set function of supermodular degree  $d$  has supermodular width at most  $d$ , and there exists a supermodular-width-1 function over a ground set of  $m$  elements whose supermodular degree is  $m - 1$ . We show that previous results regarding approximation guarantees for welfare and constrained maximization as well as regarding the Price of Anarchy (PoA) of simple auctions can be extended without any loss from the supermodular degree to the supermodular width. We also establish almost matching information-theoretical lower bounds for these two well-studied fundamental maximization problems over set functions. The combination of these approximation and hardness results illustrate that the SMW hierarchy provides not only a natural notion of complementarity, but also an accurate characterization of “near submodularity” needed for maximization approximation. While SD and SMW hierarchies support nontrivial bounds on the PoA of simple auctions, we show that our SAW hierarchy seems to capture more intrinsic properties needed to realize the efficiency of simple auctions. So far, the SAW hierarchy provides the best dependency for the PoA of Single-bid Auction, and is nearly as competitive as the Maximum over Positive Hypergraphs (MPH) hierarchy for Simultaneous Item First Price Auction (SIA). We also provide almost tight lower bounds for the PoA of both auctions with respect to the SAW hierarchy.

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

**Keywords and phrases** set functions, measure of complementarity, submodularity, subadditivity, cardinality constrained maximization, welfare maximization, simple auctions, price of anarchy

---

<sup>1</sup> Supported in part by the National Natural Science Foundation of China (Grant No. 61433014).

<sup>2</sup> Supported in part by Simons Investigator Award and by NSF grant CCF-1815254.

<sup>3</sup> Supported by NSF awards IIS-1814056 and IIS-1527434.



Digital Object Identifier 10.4230/LIPIcs.ITCS.2019.24

**Related Version** A full version of the paper is available at [4], <https://arxiv.org/abs/1805.04436>.

**Acknowledgements** We thank Vincent Conitzer for helpful feedback and discussion, and anonymous reviewers for their insightful comments and suggestions.

## 1 Introduction

For a ground set  $X = [m] = \{1, 2, \dots, m\}$ , a *set function*  $f : 2^X \rightarrow \mathbb{R}$  assigns each subset  $S \subseteq X$  a real value.<sup>4</sup> Function  $f$  is *monotone* if  $f(T) \geq f(S), \forall S \subseteq T \subseteq X$ , and *normalized* if  $f(\emptyset) = 0$ . In this paper, we will focus on normalized monotone set functions, which by definition are non-negative.

Like graphs to network analysis, set functions provide the mathematical language for many applications, ranging from combinatorial auctions (economics) to coalition formation (cooperative game theory; political science) [25, 26] to influence maximization (viral marketing) [24, 17]. Because of its exponential dimensionality, set functions – which are as rich as weighted hypergraphs – are far more expressive mathematically and challenging algorithmically than graphs [28]. However, when monotone set functions are *submodular* [22, 29], or – more generally – *complement-free* [8], algorithms with remarkable performance guarantees have been developed for various optimization, social influence, economic, and learning tasks [2, 17, 20, 3, 23].

In this paper, we study two new *degree-of-complementarity* measures of monotone set functions, and demonstrate their usefulness for several optimization and economic tasks. We prove that our complementarity measures – which are based on natural *witnesses* of complementarity – introduce hierarchies (over monotone set functions) that smoothly move beyond submodularity and subadditivity.

### 1.1 Witnesses to Complementarity: Supermodular Sets and Superadditive Sets

For any sets  $S, T \subseteq X$ , let  $f(S|T) := f(S \cup T) - f(T)$  be the *margin* of  $S$  given  $T$ . Recall that  $f$  is *subadditive* if  $f(S \cup T) \leq f(S) + f(T), \forall S, T \subseteq X$ , and *submodular* if for all  $S, T$  and  $v \in X \setminus T$ ,  $f(v|S \cup T) \leq f(v|S)$ . It is well known that every submodular set function is also subadditive. If there are sets  $S, T \subseteq V$  such that  $f(S \cup T) > f(S) + f(T)$ , then one may say that  $(S, T)$  is a witness to *complementarity* in  $f$ . Motivated by a line of recent work [1, 14, 10, 9, 11, 6], we consider the following fundamental question about set functions:

*Are there other natural, and preferably more general, forms of witnesses to complementarity that have algorithmic consequences?*

The supermodular degree of Feige and Izsak [10] is among the first measures of complementarity that are connected with algorithmic solutions to monotone-set-function maximization and combinatorial auctions. The supermodular degree is defined based on a notion of positive dependency between elements:  $u \in X$  positively depends on  $v \in X \setminus \{u\}$  (denoted by  $u \rightarrow^+ v$ ), if there exists  $S \subseteq X$  such that  $f(u|S) > f(u|S \setminus \{v\})$ .

<sup>4</sup> Throughout the paper we use  $m$  to denote the number of elements in the ground set.

► **Definition 1.1** (Supermodular Degree). The supermodular degree of a set function  $f : 2^X \rightarrow \mathbb{R}^+$ ,  $SD(f)$ , is defined to be  $SD(f) = \max_{u \in X} |\text{Dep}_f^+(u)|$ , where  $\text{Dep}_f^+(u) = \{v | u \rightarrow^+ v\}$ .

Although supermodular degree has been shown useful in a number of settings, it is not clear whether it provides the tightest possible characterization of supermodularity. For example, consider a customer who wants any two or more items out of  $m$  items, but not zero or one item. That is, the customer has a valuation function, where any subset of  $[m]$  of size at least 2 provides utility 1, and any subset of size at most 1 provides utility 0. For this function, according to Feige and Izsak’s definition, any two items depend positively on each other. In particular, any item depends positively on all other items, so the supermodular degree of this valuation function is  $m - 1$  – the largest degree possible. This seems to contradict the intuition that there is only very limited complementarity.

Below, we will provide two perspectives, with the first highlighting *supermodularity* and the second highlighting *superadditivity*. We will then study how these two complementarity measures can be used to capture the performance of basic computational solutions in optimization and auction settings where the utilities are modeled by monotone set functions. In particular, our measure of supermodularity refines supermodular degree, and avoids the kind of overestimation discussed above. Our first definition focuses on modularity:

► **Definition 1.2** (Supermodular Set). Given a normalized monotone set function  $f$  over a ground set  $X$ , a set  $T \subseteq X$  is *supermodular* w.r.t.  $f$  if:

$$\exists S \subseteq X \text{ and } v \in X \setminus T, \text{ such that: } f(v|S \cup T) > \max_{T' \subsetneq T} f(v|S \cup T'). \quad (1)$$

Note that if  $f$  is submodular, then  $f(v|S \cup T) \leq f(v|S \cup T'), \forall T' \subsetneq T$ , implying  $f$  has no supermodular set. Thus, if a set function  $f$  has a supermodular set, then it is not submodular. We say that such a set  $T$  (in Definition 1.2) *complements* item  $v$  given  $S$ , where  $S$  provides the setting that demonstrates the complementarity between  $v$  and  $T$ . In the “customer example” given after Definition 1.1, we can check that any singleton is a supermodular set, but any set with size at least two is not a supermodular set, because any single item in the set already provides all the complementarity for any other single item. A supermodular set behaves similarly to the typical example of complements, namely *complementary bundles*,<sup>5</sup> in the sense that the set as a whole provides more complement to a single item than any of its strict subsets. However, supermodular sets have richer structures while preserving the strong complementarity of such bundles, making them potentially more challenging to deal with mathematically/algorithmically than complementary bundles of a similar size.

Our next definition focuses on additivity:

► **Definition 1.3** (Superadditive Set). Given a normalized monotone set function  $f$  over a ground set  $X$ , a set  $T \subseteq X$  is *superadditive* w.r.t.  $f$  if:

$$\exists S \subseteq X \setminus T \text{ such that: } f(S|T) > \max_{T' \subsetneq T} f(S|T'). \quad (2)$$

In Definition 1.3, we say such a set  $T$  *complements* set  $S$ . Note that if  $f$  is subadditive, then for  $T' = \emptyset$ ,  $f(S|T) = f(S \cup T) - f(T) \leq (f(S) + f(T)) - f(T) = f(S) = f(S) - f(T') = f(S|T')$ , implying  $f$  does not have a superadditive set. In other words, if  $f$  has any superadditive set, then it is not subadditive.

Supermodular/superadditive sets correspond to witnesses that exhibit different kinds of complementarity. Supermodular sets are sensitive to the presence of an environment, and superadditive sets model complements to sets instead of items. The cardinality of the largest

<sup>5</sup>  $S$  is a *complementary bundle* if  $f(S) > 0$  and  $\max_{S' \subsetneq S} f(S') = 0$ .

supermodular sets or superadditive sets provides a measure of the “level of complementarity”, similar to the supermodular degree ([10]), the size of the largest bundle, and the hyperedge size ([9]) (also see Definition 1.16) in previous work.

► **Definition 1.4** (Supermodular Width). The *supermodular width* of a set function  $f$  is:

$$\text{SMW}(f) := \max\{|T| \mid T \text{ is a supermodular set w.r.t. } f\}. \quad (3)$$

► **Definition 1.5** (Superadditive Width). The *superadditive width* of a set function  $f$  is:

$$\text{SAW}(f) := \max\{|T| \mid T \text{ is a superadditive set w.r.t. } f\}. \quad (4)$$

Each measure classifies monotone set functions into a hierarchy of  $m$  levels:

► **Definition 1.6** (Supermodular Width Hierarchy (SMW- $d$ )). For any integer  $d \in \{0, \dots, m-1\}$ , a set function  $f : 2^{[m]} \rightarrow \mathbb{R}$  belongs to the first  $d$ -levels of the *supermodular width hierarchy*, denoted by  $f \in \text{SMW-}d$ , if and only if  $\text{SMW}(f) \leq d$ .

► **Definition 1.7** (Superadditive Width Hierarchy (SAW- $d$ )). For any integer  $d \in \{0, \dots, m-1\}$ , a set function  $f : 2^{[m]} \rightarrow \mathbb{R}$  belongs to the first  $d$  levels of the *superadditive width hierarchy*, denoted by  $f \in \text{SAW-}d$ , if and only if  $\text{SAW}(f) \leq d$ .

We will show that functions at level 0 of the above two hierarchies, respectively, are precisely the families of submodular and subadditive functions. In both hierarchies,  $\text{SMW-}(m-1)$  and  $\text{SAW-}(m-1)$  contains all monotone set functions over  $m$  elements. Coming back again to the customer example, we see that the utility of the customer has supermodular width of 1. Comparing to its supermodular degree of  $m-1$ , our hierarchy characterizes this utility function at a much lower level, which matches our intuition that the complementarity of this customer’s utility function should be limited. We will further show below that this difference also has significant algorithmic implications.

## 1.2 Our Results and Related Work

We now summarize the technical results of this paper. Structurally, we provide strong evidence that our definitions of supermodular/superadditive sets are natural and robust. We show that they – respectively – capture a set-theoretical gap of monotone set functions to submodularity and subadditivity. Algorithmically, we prove that our characterization based on supermodular width is *strictly stronger* than that of Feige-Izsak’s based on supermodular degree, by establishing the following:

1. For every set function  $f : 2^{[m]} \rightarrow \mathbb{R}$ ,  $\text{SD}(f) \leq \text{SMW}(f)$ , and there exists a function whose supermodular degree is much larger than its supermodular width.
2. The SMW hierarchy offers the same level of algorithmic guarantees in the maximization and auction settings as the SD hierarchy.

We will also compare both hierarchies with the MPH hierarchy of [9].

### 1.2.1 Robustness: Capturing the Set-Theoretical Gap to Submodularity/Subadditivity

We interpret the level of complementarity in our formulation of supermodular and superadditive sets from a dual perspective: We prove that they characterize the gaps from a monotone set function to submodularity and subadditivity, respectively. Our characterization uses the following definition.

► **Definition 1.8** (*d*-Scopic Submodularity). For integer  $d \geq 0$ , a normalized monotone set function  $f$  is *d*-scopic submodular if and only if:

$$f(v|T) \leq \max_{T': T' \subseteq T, |T'| \leq d} f(v|S \cup T'), \quad \forall S, T \subseteq X, v \in X \text{ satisfying } S \subseteq T, v \notin T \quad (5)$$

In Condition (5),  $\{S \cup T' | T' \subseteq T, |T'| \leq d\}$  defines a set-theoretical *neighborhood* around  $S$ . The *d*-scopic submodularity means that even if the submodular condition  $f(v|T) \leq f(v|S)$  may not hold for some  $S \subseteq T$ , it holds for some set in  $S$ 's *d*-neighborhood inside  $T$ . Thus, the parameter  $d$  provides a set-theoretical scope for examining submodularity. Similarly:

► **Definition 1.9** (*d*-scopic Subadditivity). For integer  $d \geq 0$ , a set function  $f$  is *d*-scopic subadditive if and only if:

$$f(S|T) \leq \max_{T': T' \subseteq T, |T'| \leq d} f(S|T'), \quad \forall S, T \subseteq X \text{ satisfying } S \cap T = \emptyset. \quad (6)$$

In Section 2, we prove the following two theorems.

► **Theorem 1.10** (Set-Theoretical Characterization of the SMW Hierarchy). *For any integer  $d \geq 0$  and set function  $f : 2^X \rightarrow \mathbb{R}$ ,  $f$  is  $d$ -scopic submodular if and only if  $\text{SMW}(f) \leq d$ .*

► **Theorem 1.11** (Set-Theoretical Characterization of the SAW Hierarchy). *For any integer  $d \geq 0$  and set function  $f : 2^X \rightarrow \mathbb{R}$ ,  $f$  is  $d$ -scopic subadditive if and only if  $\text{SAW}(f) \leq d$ .*

With matching supermodularity/submodularity and superadditivity/subadditivity characterization, Theorems 1.10 and 1.11 illustrate that our definitions of supermodular/superadditive sets are both natural and robust. While monotone submodular functions are all subadditive, some *d*-scopic submodular functions are not *d*-scopic subadditive. In fact, these two hierarchies are not comparable (Propositions 2.6 and 2.7): They model different aspects of complementarity that can be utilized in diverse algorithmic and economic settings.

## 1.2.2 Expressiveness: Strengthening Supermodular Degree

Supermodular sets extend positive dependency (as used in supermodular degree), which – in essence – can be viewed as a graphical approximation of supermodular sets. Thus, our characterization based on supermodular width strengthens Feige-Izsak's the characterization based on supermodular degree [10].

► **Theorem 1.12.** *Every monotone set function  $f$  with supermodular degree  $d$  has supermodular width at most  $d$  (i.e., it is  $d$ -scopic submodular). Moreover, there exists a monotone set function  $f : 2^{[m]} \rightarrow \mathbb{R}^+$  with  $\text{SMW}(f) = 1$  and  $\text{SD}(f) = m - 1$ .*

In other words, the SMW hierarchy strictly *dominates* the SD hierarchy.<sup>6</sup>

## 1.2.3 Usefulness: Algorithmic and Economic Applications

We then show, algorithmically, the SMW hierarchy – while being more expressive than the SD hierarchy – provides a complexity classification as effective as the latter (Theorems 3.2, 3.5 and 4.15). We will illustrate the usefulness of our hierarchies in algorithm and auction design with two archetypal classes of problems, *set function maximization* and

<sup>6</sup> Formally, when comparing two set-function hierarchies, say with name  $\{Y_d\}_{d \in [0, m-1]}$  and  $\{Z_d\}_{d \in [0, m-1]}$ , we say  $Y$  *dominates*  $Z$ , if for all  $d \in [0, m-1]$  and  $f$ ,  $f \in Z_d$  implies  $f \in Y_d$ .



*combinatorial auctions*, which traditionally involve measures of complementarity. Motivated by previous work [10, 14, 11, 9], we will characterize the *approximation guarantee* of polynomial-time set-function maximization algorithms and *efficiency* of simple auction protocols in terms of the complementarity level in our hierarchies. In these settings, we will compare our hierarchies with two most commonly cited complementarity hierarchies: the supermodular degree (SD) hierarchy and the Maximum over Positive Hypergraphs (MPH) hierarchy.

- *Set-Function Maximization*: We will consider both constrained and welfare maximization. The former aims to find a set of a given cardinality with maximum function value. The latter aims to allocate a set of items to  $n$  agents,<sup>7</sup> with potentially different valuations, such that the total value of all agents is maximized. As a set function has exponential dimensions in  $m$ , in both maximization problems, we assume that the input set functions are given by their value oracles.
- *Combinatorial Auctions and Simple Auction Protocols*: We will consider two well-studied simple combinatorial auction protocols: Single-bid Auction and Simultaneous Item First Price Auction (SIA). In both settings, there are multiple agents, each of which has a (potentially different) valuation function over subsets of items. The former auction protocol proceeds by asking each bidder to bid a single price, and letting bidders, in descending order of their bids, buy any available set of items paying their bid for each item. The latter simply runs first-price auctions simultaneously for all items.

#### 1.2.4 Approximation Guarantees According to Supermodular Widths

We will prove that the elegant approximability results for constrained maximization by [14] and for welfare maximization by [10] can be extended from supermodular degree to supermodular width. We obtain the same dependency (see Theorems 3.2 and 3.5) – that is,  $1 - e^{-1/(d+1)}$  and  $\frac{1}{d+2}$  respectively – on the supermodular width  $d$  as what the supermodular degree previously provides for these problems.

Because our SMW hierarchy is strictly more expressive, our upper bounds for SMW- $d$  cover strictly more monotone set functions than previous results for SD- $d$ . We will also complement our algorithmic results with nearly matching information theoretical lower bounds (see Theorems 3.3 and 3.6), for these two well-studied fundamental maximization problems. Our approximation and hardness results illustrate that the SMW hierarchy not only captures a natural notion of complementarity, but also provides an accurate characterization of the “nearly submodular property” needed by approximate maximization problems.

#### 1.2.5 Efficiency of Simple Auctions According to Superadditive/Supermodular Width

Next, we will analyze the efficiency of two well-known simple auction protocols in terms of superadditive width. To state our results and compare them with previous work, we first recall a notation from [9]:

---

<sup>7</sup> Throughout the paper we use  $n$  to denote the number of agents (whenever applicable) unless otherwise specified.



► **Definition 1.13** (Closure under Maximization). For any family of set functions  $\mathcal{F}$  over  $X$ , the closure of  $\mathcal{F}$  under maximization, denoted by  $\max(\mathcal{F})$ , is the following family of set functions:  $f \in \max(\mathcal{F})$  if and only if:

$$\exists k \in \mathbb{N}, f_1, \dots, f_k \in \mathcal{F}, \text{ s.t. } f(S) = \max_{i \in [k]} f_i(S), \forall S \subseteq X. \quad (7)$$

We will prove the following nearly tight upper and lower bounds:

► **Theorem 1.14.** *Single-bid Auction and SIA are approximately efficient for valuation functions in  $\max(\text{SAW}(d))$ , with Price of Anarchy (PoA)  $O(d \log m)$ . In addition, for any  $d > 0$ , there is an instance with SAW- $d$  valuations, where the Price of Stability (PoS) of Single-bid Auction is at least  $d + 1 - \varepsilon$  for any  $\varepsilon > 0$ , and the PoA of SIA is at least  $d$ .*

Although supermodular width strictly strengthens supermodular degree, superadditive width is not comparable with supermodular degree. Nevertheless, our PoA bound of  $O(d \log m)$  is a factor of  $d$  tighter than the  $O(d^2 \log m)$  supermodular-degree based bound of [11] for Single-bid Auction. This improvement of dependency on  $d$ , together with the nearly matching lower bound, suggests that the SAW hierarchy might be more capable in capturing the smooth transition of efficiency of simple auctions. Furthermore, as a byproduct of our efficiency results for the SAW hierarchy, we also obtain similar results, but with a worse dependency on  $d$ , for the SMW hierarchy.

► **Theorem 1.15.** *Single-bid auction and SIA are approximately efficient for valuations in  $\max(\text{SMW-}d \cap \text{SUPADD})$  – with PoA  $O(d^2 \log m)$  – where SUPADD denotes the class of monotone superadditive set functions.*

For Single-bid Auction, this result strengthens the central efficiency result of [11] by replacing the supermodular degree with the more inclusive supermodular width. For the PoA analysis of SIA, the the Maximum over Positive Hypergraphs (MPH) hierarchy of [9] remains the gold standard, by providing asymptotically matching upper and lower bounds. MPH is defined based on the following hypergraph characterization of set functions: Every normalized monotone set function over ground set  $X$  can be uniquely expressed by another set function  $h$  such that  $f(S) = \sum_{T \subseteq S} h(T)$ ,  $\forall S \subseteq X$ , where  $h(T)$  for each  $T$  is called the weight of hyperedge  $T$ .

► **Definition 1.16** (Maximum over Positive Hypergraphs [9]). Let PH- $d$  be the class of set functions whose hypergraph representation  $h$  satisfies: (1)  $h(S) \geq 0$  for all  $S$ , and (2)  $h(S) > 0$  only if  $|S| \leq d$ . The  $d$ -th level of the MPH hierarchy is:  $\text{MPH-}d = \max(\text{PH-}d)$ .

MPH provides the best characterization to the efficiency of SIA as well as ties with SD and SMW regarding the approximation ratio of welfare maximization (although it requires access to the much stronger demand oracles). However, it remains open whether it can be used to analyze constrained set function maximization and Single-bid Auction. See Table 1 for a comparison. We will prove the following theorem which states that, in general, the SAW hierarchy is not comparable with MPH.

► **Theorem 1.17.** *There is a subadditive function that lives in an upper (i.e.  $\geq m/2$ ) MPH level. On the other direction, there is a function on level 2 of MPH, whose superadditive and supermodular widths are both  $m - 1$ .*

It remains open whether  $\text{MPH-}(d + 1)$  – which subsumes  $\text{SD-}d$  as a subset – contains  $\text{SMW-}d$ . In particular, the proof that  $\text{SD-}d \subseteq \text{MPH-}(d + 1)$  in [9] does not appear easily applicable to  $\text{SMW-}d$ .

■ **Table 1** Comparison of hierarchies of complementarity. Note that the  $O(d)$  bound for PoA of SIA with SD- $d$  valuations follows from the fact that SD- $d \subseteq$  MPH- $(d+1)$ , which is not clearly comparable with the PoA bound of SIA with SMW- $d$  valuations. See corresponding references and theorems for more accurate statements.

	SD- $d$	MPH- $(d+1)$	SMW- $d$	SAW- $d$
constrained maximization	$1 - e^{1/(d+1)}$ [14]	?	$1 - e^{1/(d+1)}$ (Thm 3.2)	?
welfare maximization	$1/(d+2)$ [10]	$1/(d+2)$ [9]	$1/(d+2)$ (Thm 3.5)	?
PoA of Single-bid Auction	$O(d^2 \log m)$ [11]	?	$O(d^2 \log m)$ (Thm 4.14)	$O(d \log m)$ (Thm 4.9)
PoA of SIA	$O(d)$ [9]	$O(d)$ [9]	$O(d^2 \log m)$ (Thm 4.15)	$O(d \log m)$ (Thm 4.10)

### 1.2.6 Other Related Work

**Set Function Maximization.** There is a rich body of research focusing on set function maximization with complement-free functions, e.g. [22, 29, 8]. Various information/complexity theoretical lower bounds have been established for both problems, e.g. [21, 7, 20, 18].

**Efficiency of Simple Auctions.** Single-bid Auction with subadditive valuations has a PoA of  $O(\log m)$  [5]. SIA with subadditive valuations has a constant PoA [12]. Posted price auctions with XOS valuations give a constant factor approximate welfare guarantee [13].

**Other Measures of Complementarity.** Some other useful measures include Positive Hypergraph (PH) [1] and Positive Lower Envelop (PLE) [9]. Eden *et al.* recently introduce an extensive measure which ranges from 1 to  $2^m$  to capture the smooth transition of revenue approximation guarantee [6].

## 2 Expressiveness of the New Hierarchies

### 2.1 Characterization of Supermodular/Superadditive Widths

We first prove Theorems 1.10 and 1.11, which characterize supermodular/superadditive widths with  $d$ -scopic submodular/subadditive functions.

**Proof of Theorem 1.10.** We now show  $\text{SMW}(f) \leq d$  iff  $f$  is  $d$ -scopic submodular. First, suppose  $\text{SMW}(f) \leq d$ . Consider any triple  $(T, S, v)$  such that  $S \subseteq T \subseteq X$  and  $v \notin T$ . To show  $f$  is  $d$ -scopic submodular, we prove by induction on the size of  $T$ , that

$$f(v|T) \leq \max_{T': T' \subseteq T, |T'| \leq d} f(v|S \cup T'). \quad (8)$$

As the base case, when  $|T| \leq d$ , the inequality of (8) trivially holds because if  $T' = T \setminus S$ , then  $|T'| \leq d$  and  $f(v|S \cup T') = f(v|T)$ . Inductively, assume that the statement is true for all  $\{V \subseteq X : |V| \leq k\}$  for some  $k \geq d$ . Now consider any set  $T$  with  $|T| = k+1 > d$ . Because  $T$  is not supermodular, there is  $T'' \subsetneq T$ , such that  $f(v|T) \leq f(v|T'')$ . Applying the inductive hypothesis on  $(T'', S, v)$ , we have:

$$f(v|T'') \leq \max_{T': T' \subseteq T'', |T'| \leq d} f(v|S \cup T') \leq \max_{T': T' \subseteq T, |T'| \leq d} f(v|S \cup T').$$

Thus,  $f(v|T) \leq f(v|T'') \leq \max_{T': T' \subseteq T, |T'| \leq d} f(v|S \cup T')$ , and we have demonstrated that  $f$  is  $d$ -scopic submodular. For the other direction, we assume  $f$  is  $d$ -scopic submodular. There is no supermodular set of size larger than  $d$ , because for any  $S, T, v \notin T$  where  $|T| > d$ , there is some  $T' \subseteq T$  where  $|T'| \leq d$ , such that  $f(v|S \cup T) \leq f(v|S \cup T')$ , i.e.  $T$  is not supermodular. Therefore  $\text{SMW}(f) \leq d$ . ◀

► **Corollary 2.1.**  $f$  is submodular iff  $\text{SMW}(f) = 0$  (i.e.,  $f$  has no supermodular set).

**Proof of Theorem 1.11.** We prove  $\text{SAW}(f) \leq d$  iff  $f$  is  $d$ -scopic subadditive. Suppose  $\text{SAW}(f) \leq d$ . Consider  $S$  and  $T$  where  $S \cap T = \emptyset$ . We show  $d$ -scopic subadditivity by induction on the size of  $T$ . When  $|T| \leq d$ , the statement trivially holds. Suppose  $d$ -scopic subadditivity holds for  $|T| \leq k$  where  $k \geq d$ . For  $|T| = k+1 > d$ , since  $T$  is not superadditive, there is  $T'' \subsetneq T$ , such that  $f(S|T) \leq f(S|T'')$ . Applying inductive hypothesis on  $S, T''$  gives  $f(S|T) \leq f(S|T'') \leq \max_{T': T' \subseteq T, |T'| \leq d} f(S|T')$ , i.e.  $f$  is  $d$ -scopic subadditive.

Now assume  $d$ -scopic subadditivity. There is no superadditive set with size larger than  $d$ , because for any  $S$  and  $T$  where  $|T| > d$  and  $S \cap T = \emptyset$ , there is some  $T' \subseteq T$  where  $|T'| \leq d$ , such that  $f(S|T) \leq f(S|T')$ , i.e.  $T$  is not superadditive. ◀

► **Corollary 2.2.**  $f$  is subadditive iff  $\text{SAW}(f) = 0$  (i.e.,  $f$  has no superadditive set).

## 2.2 Supermodular Width vs Supermodular Degree

The following two propositions establish Theorem 1.12, showing supermodular width strictly dominates supermodular degree.

► **Proposition 2.3.** For any set function  $f$ ,  $\text{SD}(f) \leq \text{SMW}(f)$ .

**Proof.** Fix  $f$ . Let  $T$  be a supermodular set of size  $\text{SMW}(f)$ , and  $S, v$  be such that  $f(v|T \cup S) > f(v|T' \cup S), \forall T' \subsetneq T$ . Clearly for any  $t \in T$ ,  $f(v|\{t\} \cup (T \setminus \{t\}) \cup S) > f(v|(T \setminus \{t\}) \cup S)$ . In other words,  $v \rightarrow^+ t$  for all  $t \in T$ , so  $\text{SD}(f) \geq \text{deg}^+(v) \geq |T| = \text{SMW}(f)$ . ◀

► **Proposition 2.4.** There is a monotone set function  $f$  with  $\text{SMW}(f) = 1$  and  $\text{SD}(f) = m-1$ .

**Proof.** Consider a symmetric<sup>8</sup>  $f$  over a ground set  $X = [m]$ , where  $f(S) = 0$  if  $|S| \leq 1$ , and  $f(S) = 1$  otherwise. Observe that for any  $u \neq v$ ,  $1 = f(u|\{v\}) > f(u|\emptyset) = f(u) = 0$ , so  $u \rightarrow^+ v$ , and  $\text{SD}(f) = |\text{Dep}_f^+(u)| = m-1$ . On the other hand, consider any  $T$  where  $|T| \geq 2$ . For any  $v, S$ , we have  $|S \cup T| \geq 2$ , so  $0 = f(v|S \cup T) \leq f(v|S)$ . Thus,  $T$  is not supermodular. Since there is no supermodular set with size larger than 1 and  $f$  is not submodular,  $\text{SMW}(f) = 1$ . ◀

While the SAW hierarchy does not subsume the MPH hierarchy (see Proposition 2.8), we show that there is a monotone set function in the lowest layer of the SAW hierarchy (i.e. a subadditive function) and a notably high layer of the MPH hierarchy.

► **Proposition 2.5.** There is a monotone set function  $f$  with  $\text{SAW}(f) = 0$  and  $\text{MPH}(f) = m/2$ .

**Proof.** The proposition is a direct corollary of Proposition L.2 in [9]. ◀

<sup>8</sup>  $f$  is symmetric if  $f(S)$  depends only on  $|S|$ .

### 2.3 Further Comparisons between Hierarchies

► **Proposition 2.6.** *There is a monotone set function  $f$  with  $\text{SMW}(f) = 1$  and  $\text{SAW}(f) = m/2$ .*

**Proof.** Let  $h_T(S) = \mathbb{I}[T \subseteq S]$ . Consider  $f : 2^X \rightarrow \mathbb{R}^+$  where  $X = [2t]$  and  $f(S) = \sum_{i \in [t]} h_{\{i, i+t\}}(S)$ . Because the only complement set to any item  $i \in [t]$  is  $i + t$ ,  $\text{SMW}(f) = 1$ . Note also  $T = \{t + 1, \dots, 2t\}$  is a complement set to  $S = [t]$ , so  $\text{SAW}(f) = t = m/2$ . ◀

► **Proposition 2.7.** *There is a monotone set function  $f$  with  $\text{SAW}(f) = 0$  and  $\text{SMW}(f) = m - 1$ .*

**Proof.** Consider a symmetric  $f : 2^X \rightarrow \mathbb{R}^+$ , where  $f(\emptyset) = 0$ ,  $f(X) = 2$  and  $f(S) = 1$  otherwise.  $f$  is subadditive so  $\text{SAW}(f) = 0$ . On the other hand,  $X \setminus \{u\}$  for any  $u$  is a complement set to  $u$ , so  $\text{SMW}(f) = m - 1$ . ◀

► **Proposition 2.8.** *There exists a monotone set function  $f$  with  $\text{MPH}(f) = 2$  and  $\text{SMW}(f) = \text{SAW}(f) = m - 1$ .*

**Proof.** Let  $h_T(S) = \mathbb{I}[T \subseteq S]$ . Consider function  $f : 2^X \rightarrow \mathbb{R}^+$  where  $f(S) = \sum_{u \neq v} h_{\{u, v\}}(S)$ . Note that  $f$  is in MPH-2 since its hypergraph representation consists of only hyperedges of size 2. Now consider any  $u$  and  $T = X \setminus \{u\}$ . For any  $T' \subsetneq T$ ,  $f(u|T) = |T| > |T'| = f(u|T')$ . Thus,  $T$  is both supermodular and superadditive, and  $\text{SMW}(f) = \text{SAW}(f) = m - 1$ . ◀

## 3 Expanding Approximation Guarantees for Classic Maximization

In this section, we focus on the connection between supermodular width and two classical optimization problems: the constrained and welfare set-function maximization. For submodular functions, greedy algorithms provide tight approximation guarantees for both problems [22, 29]. Here, simple modifications to these greedy algorithms can effectively utilize the mathematical structure underlying the gap to submodularity in any set function  $f$ . These extensions achieve approximation ratios parametrized by the supermodular width with the same dependency as the supermodular degree provides [14, 10]. We complement our approximation results by nearly tight information-theoretical lower bounds.

### 3.1 Constrained Maximization

We first focus on cardinality constrained maximization, a problem at the center of resource allocation and network influence [24, 17, 22, 29]. Formally:

► **Definition 3.1** (Cardinality Constrained Maximization). Given a monotone set function  $f : 2^X \rightarrow \mathbb{R}^+ \cup \{0\}$  and integer  $k > 0$ , compute a set  $S \subseteq X$  with  $|S| \leq k$  that maximizes  $f(S)$ .

We will analyze an algorithm which performs *batched greedy selection*, – see Algorithm 1 below – where the batch size is a function of the supermodular width of  $f$ . In particular, for an input set function, the batched greedy algorithm chooses a set of size not exceeding  $\text{SMW}(f) + 1$  which maximizes marginal gain, till all  $k$  elements are chosen.

Below, we show that this simple greedy algorithm provides strong approximation guarantees in terms of the supermodular width of the input function.

► **Theorem 3.2** (Extending [14]). *For any monotone set function  $f$  over  $[m]$ , Algorithm 1 achieves  $(1 - e^{-1/(\text{SMW}(f)+1)})$ -approximation for constrained maximization problem after making  $O(m^{\text{SMW}(f)+1})$  value queries.*

---

**ALGORITHM 1:** Batched Greedy Selection for Constrained Maximization  $(f, k)$ .
 

---

```

let  $S_0 \leftarrow \emptyset$ ;  $i = 0$ ;
while  $|S_i| < k$  do
  Let  $i = i + 1$ ;  $T_i \leftarrow \operatorname{argmax}_{T' \subseteq [m], |T'| \leq s} f(T' | S_i)$  where  $s = \min\{\operatorname{SMW}(f) + 1, k - |S_{i-1}|\}$ ;
  let  $S_i \leftarrow S_{i-1} \cup T_i$ ; ;
end
return  $S^{\text{BatchedGreedy}} := S_i$ ;

```

---

**Proof.** The proof uses similar ideas to those in [14], which are originally from [22]. Let  $d = \operatorname{SMW}(f)$  and (w.l.o.g.) let  $S^* = [k] = \{1, \dots, k\}$  be an optimal solution.

$$f(S^*) - f(S_i) \leq f(S^* \cup S_i) - f(S_i) \quad (9)$$

$$\leq f(S^* | S_i) = f([k] | S_i) = \sum_{j \in [k]} f(j | [j-1] \cup S_i) \leq k \max_j f(j | [j-1] \cup S_i)$$

$$\leq k \max_j \max_{U_j: U_j \subseteq [j-1], |U_j| \leq d} f(j | U_j \cup S_i) \quad (10)$$

$$\leq k \max_j \max_{U_j: U_j \subseteq [j-1], |U_j| \leq d} f(\{j\} \cup U_j | S_i) \quad (11)$$

$$\leq k f(S_{i+1} | S_i) \quad (12)$$

$$= k(f(S_{i+1}) - f(S_i))$$

where (9) is by the monotonicity of  $f$ , (10) is by the equivalent  $d$ -scopic submodularity of  $f$ , (11) is again by the monotonicity of  $f$ , and (12) is by the greedy property:  $f(S_{i+1} | S_i) = \max_{S: |S| \leq d+1} f(S | S_i)$ .

Now we have

$$\begin{aligned} f(S^*) - f(S_i) &\leq \frac{k-1}{k} (f(S^*) - f(S_{i-1})) \leq \left(\frac{k-1}{k}\right)^i (f(S^*) - f(S_0)) \\ &= \left(\frac{k-1}{k}\right)^i f(S^*) \leq e^{-i/k} f(S^*). \end{aligned}$$

Because  $f$  is monotone, we have  $|T_i| = d + 1$ , for all intermediate steps, i.e.,  $i < \lceil \frac{k}{\operatorname{SMW}(f)+1} \rceil$ . Thus, Algorithm 1 takes exactly  $t := \lceil \frac{k}{\operatorname{SMW}(f)+1} \rceil$  steps to terminate. The function value of its output  $f(S^{\text{BatchedGreedy}}) := f(S_t) \geq (1 - e^{-1/(\operatorname{SMW}(f)+1)}) f(S^*)$ . ◀

While in general, Theorem 3.2 establishes a tighter approximation guarantee for the SMW hierarchy than that for the SD hierarchy, we note that in case of submodular degree, if the positive dependency graph is given, the running times are often of the form  $\operatorname{poly}(n) \cdot 2^{O(\operatorname{SD}(f))}$ , which can be significantly better than  $n^{O(\operatorname{SMW}(f))}$  even if the submodular width  $\operatorname{SMW}(f)$  is much smaller than the submodular degree  $\operatorname{SD}(f)$ .

We now provide a nearly-matching information-theoretical lower bound, suggesting that our approximation guarantee is essentially optimal. In the theorem below, the exponent  $k^{0.99}$  can be replaced by any function of  $k$  in  $o(k)$ .

▶ **Theorem 3.3.** *For any  $d \in \mathbb{N}$ ,  $\varepsilon > 0$ , and a large enough integer  $k$ , there exists a set function  $f : 2^{[m]} \rightarrow \mathbb{R}^+$ , with  $\operatorname{SMW}(f) = d$ , such that any (possibly randomized) algorithm that produces a  $(1/(d+1) + \varepsilon)$ -approximation (with a constant probability if randomized) for the  $k$ -constrained maximization problem makes at least  $\Omega\left((m/2k)^{k^{0.99}}\right)$  value queries.*

## 24:12 Capturing Complementarity in Set Functions

**Proof.** The proof is based on similar high-level ideas to those in [20], but the detailed construction and key properties used are different. Consider a ground set  $X$  of  $m$  elements, which contains a subset  $R$  of  $r$  “special” elements. We will specify  $r$  below. We now construct a “hard-to-distinguish” function  $f_R$  such that for any  $S \subseteq X$ ,  $f_R(S) = g_R(|S|, \mathbb{I}[R \subseteq S])$  for a function  $g_R : \mathbb{N} \times \{0, 1\} \rightarrow \mathbb{R}$ . In other words,  $f_R$  depends on the cardinality of  $S$  and whether or not  $S$  completely contains  $R$ . For discussion below, let  $D = d + 1$ , and let  $c_1$  and  $c_2$  be two integers to be determined later. We set  $|R| = r = c_1 \cdot D + 1$ . We define  $f_R$  as follows:

$$f_R(S) = \begin{cases} \lfloor |S|/D \rfloor, & |S| \leq c_1 D \\ \lfloor (|S| - c_1 D)/D \rfloor + c_1, & c_1 D < |S| \leq (c_1 + c_2)D, R \not\subseteq S \\ |S| - c_1(D - 1), & c_1 D < |S| \leq (c_1 + c_2)D, R \subseteq S \\ |S| - (c_1 + c_2)(D - 1), & (c_1 + c_2)D < |S| \leq (c_1 + c_2)D + c_2(D - 1), R \not\subseteq S \\ c_1 + c_2 D, & (c_1 + c_2)D < |S| \leq (c_1 + c_2)D + c_2(D - 1), R \subseteq S \\ c_1 + c_2 D, & (c_1 + c_2)D + c_2(D - 1) < |S| \leq m \end{cases}.$$

We will use the following three properties of  $f_R$ :

- Whenever  $|S| \bmod D = D - 1$ , for any  $v \notin S$ ,  $f_R(v|S) = 1$ . Consequently,  $\text{SMW}(f_R) \leq d, \forall R \subseteq X$  with  $|R| = r$ . In fact, this property ensures that  $f_R(v|S \cup T') \geq f_R(v|S \cup T)$ , for any  $v \in X$ ,  $S, T \subseteq X$  with  $|T| \geq D = d + 1$ , and any proper subset  $T'$  of  $T$  with  $|S \cup T'| \bmod D = D - 1$ . Note that such a subset  $T'$  always exists.
- $\max \{f_R(S) \mid |S| = (c_1 + c_2)D\} = c_1 + c_2 D$ . The maximum is achieved whenever  $R \subseteq S$ .
- For any  $S \subseteq X$  satisfying  $|S| = c_1 + c_2 D$  and  $R \not\subseteq S$ ,  $f_R(S) = c_1 + c_2$ .

First, consider  $k = (c_1 + c_2)D$ . We have, for any  $S$  with  $|S| = k$ :

$$f_R(S) = \begin{cases} c_1 + c_2 D & \text{if } R \subseteq S \\ c_1 + c_2 & \text{otherwise.} \end{cases}$$

Suppose  $c_1 = o(c_2)$ . To obtain an approximation ratio better than  $(c_1 + c_2)/(c_1 + c_2 D) \rightarrow 1/D$  for  $k$ -constrained maximization of  $f_R$ , any algorithm must find a set with size  $k$  that contains all special elements in  $R$ .

For our lower bound, we will analyze the following slightly relaxed variation of the problem: Let  $K = (c_1 + c_2)D + c_2(D - 1) - 1 > k$ . Find a set of size  $K$  which contains  $R$  as a subset. Note that  $K$  is the largest number where  $f_R(S)$  – for  $|S| = K$  – depends on whether or not  $S$  contains  $R$ . In this case, note that the algorithm has no incentive to make queries of  $f_R(S)$  for  $|S| < K$  or  $|S| > K$ , because the former reveals no more information than querying any of its supersets of size  $K$ , and the latter simply does not give any information.

We first focus on the query complexity of any deterministic optimization algorithm. Assume the algorithm makes  $T$  queries regarding  $S_1, \dots, S_T$ , where  $|S_i| = K, \forall i \in [T]$ , which are deterministically chosen when the algorithm is fixed. We now establish a condition on  $T$  such that there is a subset  $R$  such that  $R \not\subseteq S_i, \forall i \in [T]$ . Consider the distribution where the  $r$  elements are selected uniformly at random. Let  $C_i$  be the event that  $S_i$  contains  $R$ . Then,

$$\begin{aligned} \Pr[C_1 \cup \dots \cup C_T] &\leq \sum_i \Pr[C_i] < \sum_i \left(\frac{|S_i|}{m}\right)^r \\ &= T \left(\frac{(c_1 + c_2)D + (D - 1)c_2 - 1}{m}\right)^{c_1 D + 1} \leq T \left(\frac{2c_2 D}{m}\right)^{c_1 D}. \end{aligned}$$

So, if  $T \leq [m/(2c_2 D)]^{c_1 D}$  then  $\Pr[C_1 \cup \dots \cup C_T] < 1$ . In other words, for any selections  $S_1, \dots, S_T \subseteq X$  with  $|S_i| = K$ , there is a subset  $R$ , such that  $R \not\subseteq S_i, \forall i \in [T]$ , implying the deterministic algorithm with querying set  $S_1, \dots, S_T$  will not find a good approximation

---

**ALGORITHM 2:** Batched Greedy for Welfare Maximization  $(f_1, \dots, f_n)$ .
 

---

```

for  $j \in [n]$  do
  | let  $X_{j,0} \leftarrow \emptyset$ ;
end
Let  $d = \max_j \{\text{SMW}(f_j)\}$ ; let  $i = 0$ ;
while  $\cup_j X_{j,i} \neq X$  do
  | Let  $i = i + 1$ ; let
  |    $(T_i, j_i^*) = \operatorname{argmax}_{(T', j'): |T'| \leq s, j' \in [n]} f_{j'}(T' | X_{j, i-1})$  where  $s = \min \left\{ d + 1, n - \sum_j |X_{j, i-1}| \right\}$ ;
  | Let  $X_{j_i^*, i} \leftarrow X_{j_i^*, i-1} \cup T_i$ ;
  | for  $j \in [n] \setminus \{j_i^*\}$  do
  |   | let  $X_{j, i} \leftarrow X_{j, i-1}$ ;
  |   end
  | return  $X_j^{\text{BatchedGreedy}} := X_{j, i}$  for every agent  $j$ ;
end

```

---

to  $f_R$ . Let  $c_2 = \frac{1}{2}c_1^{1.01}$ , so  $k^{0.99} = ((c_1 + c_2)D)^{0.99} \leq (c_1^{1.01}D)^{0.99} \leq c_1D$ . We have  $(m/2c_2D)^{c_1D} \geq (m/2k)^{k^{0.99}}$ . Thus, we conclude that any  $(1/(d+1) + \varepsilon)$ -approximation deterministic algorithm must make at least  $(m/2k)^{k^{0.99}}$  value queries.

Now consider a randomized optimization algorithm. Conditioned on the random bits of the algorithm, the above argument still works. Taking expectation of the probability of success, we see that the overall probability of success is at most  $T(2k/m)^{k^{0.99}}$ . Thus, a constant probability of success requires  $T = \Omega\left((m/2k)^{k^{0.99}}\right)$ . ◀

## 3.2 Welfare Maximization

We now turn our attention to welfare maximization. Formally:

► **Definition 3.4** (Welfare Maximization). Given  $n$  monotone set functions  $f_1, \dots, f_n$  over  $2^{[m]}$ , compute  $n$  disjoint sets  $X_1, \dots, X_n$  that maximizes  $\sum_{i \in [n]} f_i(X_i)$ .

Because  $f_1, \dots, f_n$  are monotone, the optimal solution to welfare maximization is a partition of  $X = [m]$ . Thus, welfare maximization can also be viewed as a generalized clustering or multiway partitioning problem.

We will analyze the following greedy algorithm – see Algorithm 2 below – which repeatedly assigns groups of elements to agents. At each step, the algorithm picks a set of size not exceeding  $\max_i \text{SMW}(f_i) + 1$  – as opposed to one – that provides the largest possible marginal gain to some agent and assigns the set to that agent.

We now prove the following approximation guarantee in terms of supermodular width.

► **Theorem 3.5** (Extending [10]). *For any collection of monotone set functions  $f_1, \dots, f_n$  over  $X = [m]$ , Algorithm 2 achieves  $\frac{1}{2 + \max_i \{\text{SMW}(f_i)\}}$ -approximation for welfare maximization, after making  $O\left(nm^{\max_i \{\text{SMW}(f_i)\} + 1}\right)$  value queries.*

The proof uses similar ideas to those in [10], which are originally from [16]. Due to space limit, we relegate the proof to the full version of the paper [4].

To show that our algorithm is nearly optimal, we prove the following information-theoretical lower bound: Similar to Theorem 3.3, the exponent  $(m/n)^{0.99}$  in the theorem below, can be replaced by any function of  $m/n$  in  $o(m/n)$ .



► **Theorem 3.6.** *For any  $d \in \mathbb{N}$ ,  $\varepsilon > 0$ , there is a family of function  $f_1, \dots, f_n : 2^{[m]} \rightarrow \mathbb{R}^+$  with  $\text{SMW}(f_i) = d, \forall i \in [n]$ , such that any (possibly randomized) algorithm that produces a  $(1/(d+1) + \varepsilon)$ -approximation (with constant probability if randomized) for the  $n$ -agent welfare maximization problem makes at least  $\Omega\left((n/2D)^{(m/n)^{0.99}}\right)$  value queries.*

The proof follows from a similar argument as the proof for Theorem 3.3. Due to space limit, we relegate the proof to the full version of the paper.

## 4 Efficiency of Simple Auctions

In this section, we study the connection between the SAW hierarchy and efficiency of auctions. We will draw extensively on previous work in this area, particularly on the characterization based on the *CH hierarchy* – see definition below – which is arguably the most simple class of set functions with complementarity.

► **Definition 4.1** ( *$d$ -Constraint Homogeneous Functions [11]*). A set function  $f$  over ground set  $X$  is  $d$ -constraint homogeneous (CH- $d$ ) if there exists a value  $\hat{f}$ , and disjoint sets  $Q_1, \dots, Q_h \subseteq X$  with  $|Q_i| \leq d, \forall i \in [h]$ , such that (1)  $f(Q_i) = \hat{f} \cdot |Q_i|, \forall i \in [h]$ , and (2) the value of every set  $S \subseteq [m]$  is simply the sum of values of contained  $Q_i$ 's, i.e.,  $f(S) = \sum_{Q_i \subseteq S} f(Q_i) = \hat{f} \cdot \sum_{Q_i \subseteq S} |Q_i|$ .

We will show that previous characterization of auction efficiency [11] can be approximately extended from the CH hierarchy to the SAW hierarchy.

### 4.1 Backgrounds: Related Definitions and Results

We first restate a useful definition and a lemma for analyzing the efficiency of auction mechanisms.

► **Definition 4.2** ([27]). An auction mechanism  $\mathcal{M}$  is  $(\lambda, \mu)$ -smooth for a class of valuations  $\mathcal{F} = \times_i \mathcal{F}_i$  if for any valuation profile  $f \in \mathcal{F}$ , there exists a (possibly randomized) action profile  $a_i^*(f)$  such that for every action profile  $a$ :

$$\sum_i \mathbb{E}_{a_i' \sim a_i^*(f)} [u_i(a_i', a_{-i}; f_i)] \geq \lambda \cdot \text{OPT}(f) - \mu \sum_i P_i(a),$$

where  $u_i(a_i'; f_i)$  is the utility of  $i$  given action profile  $(a_i', a_{-i})$ ,  $\text{OPT}(f)$  is the optimum social welfare given valuation profile  $f$ , and  $P_i(a)$  is the payment of  $i$  given action profile  $a$ .

► **Lemma 4.3** ([27]). *If a mechanism is  $(\lambda, \mu)$ -smooth then the price of anarchy w.r.t. coarse correlated equilibria is at most  $\max\{1, \mu\}/\lambda$ .*

For Single-bid Auction and Simultaneous Item First Price Auction (SIA), we will derive our results from the following results for CH- $d$  and MPH- $d$ .

► **Theorem 4.4** (Smoothness of Single-bid Auction with CH- $d$  Valuations [11]). *Single-bid Auction is a  $((1 - e^{-d})/d, 1)$ -smooth mechanism when agents have CH- $d$  valuations. Consequently, Single-bid Auction has a PoA of  $(1 - e^{-d})/d$  with CH- $d$  valuations w.r.t. coarse correlated equilibria.*

► **Theorem 4.5** (Smoothness of SIA with MPH- $d$  Valuations [9]). *For SIA, when bidders have MPH- $d$  valuations, both the correlated price of anarchy and the Bayes-Nash price of anarchy are at most  $2d$ . The bound follows from a smoothness argument.*



A key concept to extend these results to other valuation classes is the following notion of pointwise approximation defined in [5].

► **Definition 4.6** (Pointwise Approximation [5]). A class of set functions  $\mathcal{F}$  over ground set  $X$  is pointwise  $\beta$ -approximated by another class  $\mathcal{F}'$  of set functions over  $X$  if  $\forall f \in \mathcal{F}, S \subseteq X, \exists f'_S \in \mathcal{F}'$  such that (1)  $\beta f'_S(S) \geq f(S)$  and (2)  $\forall T \subseteq X, f'_S(T) \leq f(T)$ .

For example:

► **Proposition 4.7** ([11]). *The class  $\max(\mathcal{F})$  is pointwise 1-approximated by the class  $\mathcal{F}$ .*

We say a function  $f' : 2^X \rightarrow \mathbb{R}$  pointwise  $\beta$ -approximates  $f : 2^X \rightarrow \mathbb{R}$  (at  $X$ ), if (1)  $\beta f'(X) \geq f(X)$ , and (2)  $\forall T \subseteq X, f'(T) \leq f(T)$ .

The following lemma of [5] provides a way to translate PoA bounds between classes via pointwise approximation.

► **Lemma 4.8** (Extension Lemma [5]). *If a mechanism for a combinatorial auction setting is  $(\lambda, \mu)$ -smooth for the class of set functions  $\mathcal{F}'$ , and  $\mathcal{F}$  is pointwise  $\beta$ -approximated by  $\mathcal{F}'$ , then it is  $(\frac{\lambda}{\beta}, \mu)$ -smooth for the class  $\mathcal{F}$ . And as a result, if a mechanism for a combinatorial auction setting has a PoA of  $\alpha$  given by a smoothness argument for the class  $\mathcal{F}'$ , and  $\mathcal{F}$  is pointwise  $\beta$ -approximated by  $\mathcal{F}'$ , then it has a PoA of  $\alpha\beta$  for the class  $\mathcal{F}$ .*

## 4.2 Efficiency of Simple Auctions Parametrized by SAW

Applying Lemma 4.8, we are able to translate Theorems 4.4 and 4.5 to the SAW hierarchy.

► **Theorem 4.9** (Efficiency of Single-bid Auction with SAW- $d$  Valuations). *When agents have valuations  $f_1, \dots, f_n \in \max(\text{SAW-}d)$ , Single-bid Auction has a price of anarchy of at most  $\frac{2d}{1-e^{-2d}} \cdot H_{\frac{m}{2d}}$  w.r.t. coarse correlated equilibria.*

► **Theorem 4.10** (Efficiency of SIA with SAW- $d$  Valuations). *When agents have valuations  $f_1, \dots, f_n \in \max(\text{SAW-}d)$ , SIA has a price of anarchy of at most  $8d \cdot H_{\frac{m}{2d}}$  w.r.t. coarse correlated equilibria.*

Formally, Theorems 4.9 and 4.10 follow from Theorems 4.4 and 4.5 respectively, with the help of Lemma 4.8, Proposition 4.7, and the technical lemma (Lemma 4.11) that we will establish below, showing that for any  $d \in \mathbb{N}$ , functions in SAW- $d$  can be approximated by CH- $2d$  functions. In particular, Lemma 4.11 establishes the approximation of SAW hierarchy by CH hierarchy with a loss of factor  $O(\log m)$ .

► **Lemma 4.11** (Pointwise Approximation of SAW Hierarchy by CH-Hierarchy). *For any  $d \in \mathbb{N}$ , SAW- $d$  is pointwise  $2H_{\frac{m}{2d}}$ -approximated by CH- $2d$ , where  $H_i = \sum_{k \in [i]} \frac{1}{k}$  is the  $i$ -th harmonic number.*

**Proof.** Our proof is inspired by the constructions of [5] and [11].

For any  $f \in \text{SAW-}d$  over  $X = [m]$ , we first apply the following greedy construction to obtain a partition  $\mathcal{Q} = \{Q_i\}_{i \in [q]}$  of  $[m]$  into sets of size not exceeding  $2d$ : At step  $i$ , we select a new set  $Q_i \subseteq [m] \setminus (Q_1 \cup \dots \cup Q_{i-1})$ , with maximum  $f(Q_i)$ , among all sets of size at most  $2d$ .

We first prove by contradiction that there exists a function  $g$  in CH- $2d$  which  $2H_{\frac{m}{2d}}$ -approximates  $f$  at  $[m]$ . That is, (1)  $2H_{\frac{m}{2d}} g([m]) \geq f([m])$  and (2)  $\forall T \subseteq [m], g(T) \leq f(T)$ .

Suppose this statement is not true. Let

$$h_{\mathcal{Q}}(T) = \frac{f([m])}{\beta \cdot |\cup_i Q_i|} \sum_{Q_i \subseteq T} |Q_i|.$$

## 24:16 Capturing Complementarity in Set Functions

Note that  $h_{\mathcal{Q}} \in \text{CH-}2d$  because  $|Q_i| \leq 2d, \forall Q_i \in \mathcal{Q}$ . We now construct a series of functions based on  $h_{\mathcal{Q}}$ , and prove that for any  $\beta > 0$ , if there is no  $g$  among these functions that is a  $\beta$ -approximation of  $f$  at  $[m]$  – that is, there is no  $g$  such that (1)  $\beta g([m]) \geq f([m])$  and (2)  $\forall T \subseteq [m], g(T) \leq f(T)$ , (below we will refer to this condition as Assumption (\*)) – then  $\beta < 2H_{\frac{m}{2d}}$ .

First consider  $h_{\mathcal{Q}}$ . Note that  $\beta h_{\mathcal{Q}}([m]) = \beta \frac{f([m])}{\beta} \geq f([m])$ , because  $\mathcal{Q}$  is a partition of  $[m]$ . Assumption (\*) then implies there is a  $T_1$  such that  $h_{\mathcal{Q}}(T_1) > f(T_1)$ . W.l.o.g. assume  $T_1$  is a union of sets from  $\mathcal{Q}$  (such  $T_1$  exists because  $f$  is monotone).

Let  $S_1 = [m]$ . We now iteratively define  $S_i = S_{i-1} \setminus T_{i-1}$ , and construct its associated  $T_i$ . The construction maintains the following invariant: Both  $S_i$  and  $T_i$  are unions of sets from  $\mathcal{Q}$ . The former follows directly from the iterative property that  $S_{i-1}$  and  $T_{i-1}$  are both unions of sets from  $\mathcal{Q}$ . Our construction below will ensure the latter.

Let  $\mathcal{Q}_{S_i} = \{Q \in \mathcal{Q} \mid Q \subseteq S_i\}$ . Let

$$h_{\mathcal{Q}_{S_i}} = \frac{f([m])}{\beta \cdot |\cup_{j: Q_j \in \mathcal{Q}_{S_i}} Q_j|} \sum_{j: Q_j \in \mathcal{Q}_{S_i}} |Q_j|.$$

Again,  $h_{\mathcal{Q}_{S_i}} \in \text{CH-}2d$ , and  $h_{\mathcal{Q}_{S_i}}([m]) = \frac{f([m])}{\beta}$ . Assumption (\*) then implies there is a  $T_i$  such that  $h_{\mathcal{Q}_{S_i}}(T_i) > f(T_i)$ . Again, w.l.o.g. assume  $T_i$  is a union of sets from  $\mathcal{Q}$  (such  $T_i$  exists because  $f$  is monotone). This iterative process terminates, producing a partition  $\{T_i\}_{i \in [t]}$  of  $[m]$ , which satisfies:

$$\sum_i f(T_i) < \sum_i h_{\mathcal{Q}_{S_i}}(T_i) = \frac{f([m])}{\beta} \sum_i \frac{|T_i|}{|S_i|} \leq \frac{f([m])}{\beta} \sum_{i \in [t]} \frac{1}{i} \leq \frac{f([m])}{\beta} H_{\frac{m}{2d}}.$$

We now show that  $\sum_i f(T_i) \geq \frac{1}{2} f([m])$ . Recall that each member in partition  $\{T_i\}_i$  is a unions of sets from  $\mathcal{Q}$ . We renumber  $\{T_i\}_i$ , in a way that for any  $i < j$ , there is some  $T_i \supseteq Q_k \in \mathcal{Q}$ , such that for any  $T_j \supseteq Q_l \in \mathcal{Q}$ ,  $k < l$ . That is, the smallest index  $k$  where  $Q_k \in T_i$  is smaller than the smallest index  $l$  where  $Q_l \in T_j$ , as long as  $i < j$ .

Since  $(T_1, \dots, T_t)$  is a partition of  $[m]$ , we have:

$$\begin{aligned} f([m]) &= \sum_i f(T_i | T_{i+1} \cup \dots \cup T_t) \\ &\leq \sum_i \max\{f(T_i | U_i) \mid U_i \subseteq T_{i+1} \cup \dots \cup T_t, |U_i| \leq d\} \end{aligned} \quad (13)$$

$$\leq \sum_i \max\{f(T_i \cup U_i) \mid U_i \subseteq T_{i+1} \cup \dots \cup T_t, |U_i| \leq d\} \quad (14)$$

$$= \sum_i \max\{(f(U_i | T_i) + f(T_i)) \mid U_i \subseteq T_{i+1} \cup \dots \cup T_t, |U_i| \leq d\}$$

$$\leq \sum_i \max\{(f(U_i | V_i) + f(T_i)) \mid U_i \subseteq T_{i+1} \cup \dots \cup T_t, |U_i| \leq d, V_i \subseteq T_i, |V_i| \leq d\} \quad (15)$$

$$\leq \sum_i \max\{(f(U_i \cup V_i) + f(T_i)) \mid U_i \subseteq T_{i+1} \cup \dots \cup T_t, |U_i| \leq d, V_i \subseteq T_i, |V_i| \leq d\} \quad (16)$$

$$\leq \sum_i (f(Q_{k_i}) + f(T_i)), \text{ where } k_i = \min\{k \mid T_i \supseteq Q_k \in \mathcal{Q}\} \quad (17)$$

$$\leq \sum_i 2f(T_i), \quad (18)$$

where (13) and (15) follow from  $d$ -scopic subadditivity of  $f$ , (14), (16) and (18) follow from monotonicity of  $f$ , and (17) holds because, according to the construction of  $\{Q_l\}_l$ ,  $Q_{k_i}$

maximizes  $f$  among all sets of size  $2d$  contained in  $Q_{k_i} \cup \dots \cup Q_q \supseteq T_i \cup \dots \cup T_t$ , and in particular  $U_i \cup V_i \subseteq T_i \cup \dots \cup T_t$ .

Consequently, it follows from  $\sum_i f(T_i) \geq \frac{1}{2}f([m])$  that:

$$\frac{H_{\frac{m}{2d}} f([m])}{\beta} > \sum_i f(T_i) \geq \frac{1}{2}f([m]) \Rightarrow \beta < 2H_{\frac{m}{2d}}.$$

Thus, Assumption (\*) with  $\beta \geq 2H_{\frac{m}{2d}}$  leads to a contradiction. Therefore, we have established that there exists a CH- $2d$  function  $g$  such that (1)  $g([m]) \geq 2H_{\frac{m}{2d}}f([m])$  and (2)  $\forall T \subseteq [m]$ ,  $g(T) \leq f(T)$ .

As in [11], the above proof can be simply extended to prove for any  $S \subseteq X$ , there exists a CH- $2d$  function  $g$  such that (1)  $g(S) \geq 2H_{\frac{m}{2d}}f([m])$  and (2)  $\forall T \subseteq [m]$ ,  $g(T) \leq f(T)$ . Essentially, we restrict the function  $f$  to  $2^S$ , apply the argument above, and then span the obtained function back to  $2^X$ .

Therefore, SAW- $d$  is pointwise  $2H_{\frac{m}{2d}}$ -approximated by CH- $2d$ .  $\blacktriangleleft$

We further analyze previously known hard instances to both auctions, and show that they provide almost matching lower bounds to the above two efficiency upper bounds.

► **Theorem 4.12.** *There is an instance with SAW- $d$  valuations for any  $d$ , where the PoS of Single-bid Auction is at least  $d + 1 - \varepsilon/d$  for any  $\varepsilon > 0$ .*

► **Theorem 4.13.** *There is an instance with SAW- $d$  valuations for any  $d$ , where the PoA of SIA is at least  $d + 1/(d + 1)$ .*

We defer the proofs to the full version of the paper.

### 4.3 Efficiency of Simple Auctions Parametrized by SMW

As a byproduct of our efficiency results for the SAW hierarchy, we prove similar, but slightly weaker, results for the SMW hierarchy. We note that these bounds extend a central result in [11], which states that when agents have valuations in  $\max(\text{SD-}d \cap \text{SUPADD})$ , Single-bid Auction has a PoA of  $O(d^2 \log m)$ .

► **Theorem 4.14** (Extending [11]). *When agents have valuations  $f_1, \dots, f_n \in \max(\text{SMW-}d \cap \text{SUPADD})$ , Single-bid Auction has a price of anarchy of at most  $\frac{(d+1)^2}{1-e^{-(d+1)}} \cdot H_{\frac{m}{d+1}}$  w.r.t. coarse correlated equilibria.*

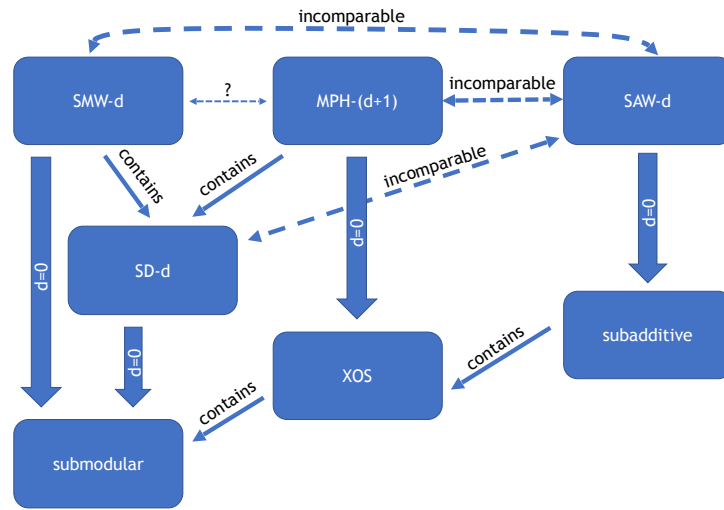
► **Theorem 4.15.** *When agents have valuations  $f_1, \dots, f_n \in \max(\text{SMW-}d \cap \text{SUPADD})$ , SIA has a price of anarchy of at most  $2(d + 1)^2 \cdot H_{\frac{m}{d+1}}$  w.r.t. coarse correlated equilibria.*

Due to space limit, we relegate the proofs to the full version of the paper.

## 5 Remarks

### 5.1 Further Comparative Analysis

As observed by Eden *et al.* [6], the right measure of complementarity often varies from application to application. This seems to be true even with the supermodular vs superadditive widths. We note that while the SD and SMW hierarchies give nontrivial bounds on the PoA of simple auctions, SAW hierarchy seems to capture the intrinsic property needed by efficiency guarantees for simple auctions. It provides tighter characterization of PoA with a gap of



■ **Figure 1** Relationship between hierarchies.

$\log m$  (instead of  $d \log m$ ) between upper and lower bounds. On the other hand, while SMW hierarchy captures the intrinsic property needed by the constrained/welfare maximization, it remains open whether a small superadditive width provides any approximation guarantee for the two optimization problems.

The MPH hierarchy takes a different approach from ours – it relies on a syntactic definition which provides elegant and intuitive structures. In contrast, both SMW and SAW hierarchies – like the SD hierarchy before it – are built on concrete natural concepts of witnesses and semantic intuition of complementarity. In the current definition, the MPH hierarchy is not an extension to submodularity or subadditivity. Rather – as shown in [9] – MPH can be considered as an extension to the fractionally subadditive (or XOS) class proposed in [19]. We therefore consider SMW, MPH and SAW parallel measures of complementarity, just like submodularity, fractional subadditivity and subadditivity in the complement-free case. One key difference is that the three hierarchies seem to diverge at higher levels of complementarity, as opposed to the fact that submodular functions are all fractionally subadditive, and fractionally subadditive functions are all subadditive. This phenomenon provides further evidence that the three hierarchies are likely to capture different aspects of complementarity. See Figure 1 for a comparison.

We also note that all upper bounds supported by our hierarchies are accompanied by almost matching lower bounds, which we consider as a justification of our definitions – they manage to categorize set functions roughly according to their “hardness” in different settings (i.e. optimization for SMW and efficiency for SAW). In contrast, while the less inclusive supermodular degree hierarchy supports a number of upper bounds, to our knowledge, none of those results are proven tight.

## 5.2 Final Remarks and Open Problems

Our SMW and SAW hierarchies may be applied to other problem settings. For example, for the online secretary problem based on supermodular degree [15], we believe that with a slight modification of the algorithms and the analysis, we could replace supermodular degree with supermodular width as well for this problem; also, SMW- $d$  functions are efficiently PAC-learnable under product distributions [30]. It may be possible to look into other venues where SMW and SAW hierarchies are applicable.

There are also a few technical questions to be answered:

- Does MPH- $(d + 1)$  – which subsumes SD- $d$  – include all SMW- $d$  functions?
- Can we improve the SAW-based efficiency characterization of Single-bid Auction and SIA to  $O(d)$ ?
- Can the MPH hierarchy be used to characterize constrained set function maximization?

---

## References

- 1 Ittai Abraham, Moshe Babaioff, Shaddin Dughmi, and Tim Roughgarden. Combinatorial auctions with restricted complements. In *Proceedings of the Thirteenth ACM Conference on Electronic Commerce*, pages 3–16. ACM, 2012.
- 2 Maria-Florina Balcan and Nicholas JA Harvey. Learning submodular functions. In *Proceedings of the Forty-Third Annual ACM Symposium on Theory of Computing*, pages 793–802. ACM, 2011.
- 3 Shuchi Chawla, Jason D. Hartline, and Robert Kleinberg. Algorithmic Pricing via Virtual Valuations. In *Proceedings of the Eighth ACM Conference on Electronic Commerce*, pages 243–251. ACM, 2007.
- 4 Wei Chen, Shang-Hua Teng, and Hanrui Zhang. Capturing Complementarity in Set Functions by Going Beyond Submodularity/Subadditivity. *arXiv preprint*, 2018. [arXiv: 1805.04436](https://arxiv.org/abs/1805.04436).
- 5 Nikhil Devanur, Jamie Morgenstern, Vasilis Syrgkanis, and S Matthew Weinberg. Simple auctions with simple strategies. In *Proceedings of the Sixteenth ACM Conference on Economics and Computation*, pages 305–322. ACM, 2015.
- 6 Alon Eden, Michal Feldman, Ophir Friedler, Inbal Talgam-Cohen, and S Matthew Weinberg. A Simple and Approximately Optimal Mechanism for a Buyer with Complements. In *Proceedings of the 2017 ACM Conference on Economics and Computation*, pages 323–323. ACM, 2017.
- 7 Uriel Feige. A threshold of  $\ln n$  for approximating set cover. *Journal of the ACM (JACM)*, 45(4):634–652, 1998.
- 8 Uriel Feige. On maximizing welfare when utility functions are subadditive. *SIAM Journal on Computing*, 39(1):122–142, 2009.
- 9 Uriel Feige, Michal Feldman, Nicole Immorlica, Rani Izsak, Brendan Lucier, and Vasilis Syrgkanis. A unifying hierarchy of valuations with complements and substitutes. In *Twenty-Ninth AAAI Conference on Artificial Intelligence*, 2015.
- 10 Uriel Feige and Rani Izsak. Welfare maximization and the supermodular degree. In *Proceedings of the Fourth Conference on Innovations in Theoretical Computer Science*, pages 247–256. ACM, 2013.
- 11 Michal Feldman, Ophir Friedler, Jamie Morgenstern, and Guy Reiner. Simple mechanisms for agents with complements. In *Proceedings of the 2016 ACM Conference on Economics and Computation*, pages 251–267. ACM, 2016.
- 12 Michal Feldman, Hu Fu, Nick Gravin, and Brendan Lucier. Simultaneous auctions are (almost) efficient. In *Proceedings of the Forty-Fifth Annual ACM Symposium on Theory of Computing*, pages 201–210. ACM, 2013.
- 13 Michal Feldman, Nick Gravin, and Brendan Lucier. Combinatorial auctions via posted prices. In *Proceedings of the Twenty-Sixth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 123–135. Society for Industrial and Applied Mathematics, 2015.
- 14 Moran Feldman and Rani Izsak. Constrained Monotone Function Maximization and the Supermodular Degree. In *Seventeenth International Workshop on Approximation Algorithms for Combinatorial Optimization Problems*, 2014.

- 15 Moran Feldman and Rani Izsak. Building a good team: Secretary problems and the supermodular degree. In *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1651–1670. SIAM, 2017.
- 16 Marshall L Fisher, George L Nemhauser, and Laurence A Wolsey. An analysis of approximations for maximizing submodular set functions—II. In *Polyhedral Combinatorics*, pages 73–87. Springer, 1978.
- 17 David Kempe, Jon Kleinberg, and Éva Tardos. Maximizing the spread of influence through a social network. In *Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 137–146. ACM, 2003.
- 18 Subhash Khot, Richard J Lipton, Evangelos Markakis, and Aranyak Mehta. Inapproximability results for combinatorial auctions with submodular utility functions. In *International Workshop on Internet and Network Economics*, pages 92–101. Springer, 2005.
- 19 Benny Lehmann, Daniel Lehmann, and Noam Nisan. Combinatorial auctions with decreasing marginal utilities. *Games and Economic Behavior*, 55(2):270–296, 2006.
- 20 Vahab Mirrokni, Michael Schapira, and Jan Vondrák. Tight information-theoretic lower bounds for welfare maximization in combinatorial auctions. In *Proceedings of the Ninth ACM Conference on Electronic Commerce*, pages 70–77. ACM, 2008.
- 21 George L Nemhauser and Laurence A Wolsey. Best algorithms for approximating the maximum of a submodular set function. *Mathematics of Operations Research*, 3(3):177–188, 1978.
- 22 George L Nemhauser, Laurence A Wolsey, and Marshall L Fisher. An analysis of approximations for maximizing submodular set functions-I. *Mathematical Programming*, 14(1):265–294, 1978.
- 23 Noam Nisan and Amir Ronen. Algorithmic Mechanism Design (Extended Abstract). In *Proceedings of the Thirty-first Annual ACM Symposium on Theory of Computing*, pages 129–140. ACM, 1999.
- 24 Matthew Richardson and Pedro Domingos. Mining Knowledge-sharing Sites for Viral Marketing. In *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 61–70, 2002.
- 25 H. E. Scarf. The core of an N person game. *Econometrica*, 69:35–50, 1967.
- 26 L. S. Shapley. On balanced sets and cores. *Naval Res. Logist. Quarter.*, 14, 1967.
- 27 Vasilis Syrgkanis and Eva Tardos. Composable and efficient mechanisms. In *Proceedings of the Forty-Fifth Annual ACM Symposium on Theory of Computing*, pages 211–220. ACM, 2013.
- 28 Shang-Hua Teng. Network Essence: PageRank Completion and Centrality-Conforming Markov Chains. In M. Loebl, J. Nešetřil, and R. Thomas, editors, *Journey Through Discrete Mathematis: A Tribute to Jiří Matoušek*. Springer, 2017.
- 29 Jan Vondrák. Optimal approximation for the submodular welfare problem in the value oracle model. In *Proceedings of the Fortieth Annual ACM Symposium on Theory of Computing*, pages 67–74. ACM, 2008.
- 30 Hanrui Zhang. Learning Set Functions with Limited Complementarity. In *Proceedings of the Thirty-Third AAAI Conference on Artificial Intelligence*, 2019.

# Probabilistic Checking Against Non-Signaling Strategies from Linearity Testing

**Alessandro Chiesa**

UC Berkeley, Berkeley, CA, USA  
alexch@berkeley.edu

**Peter Manohar**

UC Berkeley, Berkeley, CA, USA  
manohar@berkeley.edu

**Igor Shinkar**

Simon Fraser University, Vancouver, Canada  
igor.shinkar@sfu.ca

---

## Abstract

Non-signaling strategies are a generalization of quantum strategies that have been studied in physics over the past three decades. Recently, they have found applications in theoretical computer science, including to proving inapproximability results for linear programming and to constructing protocols for delegating computation. A central tool for these applications is probabilistically checkable proofs (PCPs) that are *sound against non-signaling strategies*.

In this paper we prove that the exponential-length constant-query PCP construction due to Arora et al. (JACM 1998) is sound against non-signaling strategies.

Our result offers a new length-vs-query tradeoff when compared to the non-signaling PCP of Kalai, Raz, and Rothblum (STOC 2013 and 2014) and, moreover, may serve as an intermediate step to a proof of a non-signaling analogue of the PCP Theorem.

**2012 ACM Subject Classification** Theory of computation → Computational complexity and cryptography

**Keywords and phrases** probabilistically checkable proofs, linearity testing, non-signaling strategies

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2019.25

**Related Version** Full version is available on the Electronic Colloquium on Computational Complexity as TR18-123, <https://eccc.weizmann.ac.il/report/2018/123/>.

**Funding** This work was supported by the UC Berkeley Center for Long-Term Cybersecurity.

**Acknowledgements** We are grateful to Thomas Vidick for suggestions that have improved the presentation in this paper.

## 1 Introduction

Probabilistically Checkable Proofs (PCPs) [3, 11, 2, 1] are proofs whose validity can be checked by a probabilistic verifier that accesses only a few locations of the proof. PCPs have numerous applications across the theory of computing, including to hardness of approximation [11] and delegation of computation [19, 21].



© Alessandro Chiesa, Peter Manohar, and Igor Shinkar;  
licensed under Creative Commons License CC-BY  
10th Innovations in Theoretical Computer Science (ITCS 2019).  
Editor: Avrim Blum; Article No. 25; pp. 25:1–25:17



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

A seminal result, known as the PCP Theorem [2, 1], states that every language in  $\text{NTIME}(T)$  can be probabilistically checked by a verifier that uses  $O(\log T)$  random bits and makes  $O(1)$  queries to a proof of length  $\text{poly}(T)$ .<sup>1</sup>

In this paper we study PCPs that are sound against *non-signaling strategies* (nsPCPs). These have recently found applications that appear out of the reach of (standard) PCPs, including 1-round delegation of computation from falsifiable assumptions [15, 17] and hardness of approximation for linear programming [16]. The efficiency measures achieved in known nsPCPs appear suboptimal, which affects the quality of the corresponding applications. We thus ask whether a non-signaling analogue of the PCP Theorem holds.

Below we explain the aforementioned notions, and then present our results in this direction.

**Non-signaling strategies.** Non-signaling strategies are a class of “non-local” correlations that strictly generalize quantum strategies, and capture the minimal condition that spatially-isolated parties cannot communicate instantaneously. They have been studied in physics for over three decades [24, 18, 23] in order to better understand quantum entanglement.

There are two definitions, corresponding to whether the strategy is meant to represent a function or isolated parties; the former is the relevant one for nsPCPs [15, 17].<sup>2</sup> Given a locality parameter  $k \in \mathbb{N}$ , a *k-non-signaling function*  $\mathcal{F}$  extends the notion of a function  $f: D \rightarrow \{0, 1\}$  as follows: it is a collection  $\{\mathcal{F}_S\}_{S \subseteq D, |S| \leq k}$  where each  $\mathcal{F}_S$  is a *distribution* over  $\{0, 1\}^S$  and, for every two subsets  $S_1$  and  $S_2$  each of size at most  $k$ , the restrictions of  $\mathcal{F}_{S_1}$  and  $\mathcal{F}_{S_2}$  to  $S_1 \cap S_2$  are equal as distributions.<sup>3</sup> Note that if  $k = |D|$  then  $\mathcal{F}$  is a distribution over functions  $f: D \rightarrow \{0, 1\}$ .

Note that *k-non-signaling functions* are solutions to the linear program arising from the *k-relaxation* in the Sherali–Adams hierarchy [25]. The variables are of the form  $X_{S, \vec{b}}$  (for all  $S \subseteq D$  of size at most  $k$  and  $\vec{b} \in \{0, 1\}^S$ ) and express the probability of  $\vec{b}$  in the distribution  $\mathcal{F}_S$ ; consistency across subsets  $S$  and  $T$  is expressed using the natural linear constraints.<sup>4</sup>

**Non-signaling PCPs.** Recall that a classical PCP verifier is given oracle access to a proof represented as a function  $f: D \rightarrow \{0, 1\}$ . The verifier uses random bits, makes a few queries to  $f$ , and then accepts or rejects. Completeness requires that if the statement being checked is true then there is a function  $f$  that makes the verifier always accept. Soundness requires that if the statement being checked is false then every function  $f$  makes the verifier reject with high probability.

In the non-signaling setting, “proofs” are non-signaling functions rather than (classical) functions. Soundness is correspondingly stronger: given a locality parameter  $k \in \mathbb{N}$ , soundness requires that every *k-non-signaling function*  $\mathcal{F}$  makes the nsPCP verifier reject with high probability.

<sup>1</sup> In particular, for every language in  $\text{NEXP} = \cup_{c \in \mathbb{N}} \text{NTIME}(2^{n^c})$ , the verifier uses  $\text{poly}(n)$  random bits and makes  $O(1)$  queries to a proof of length  $2^{\text{poly}(n)}$ .

<sup>2</sup> The other definition underlies the notion of multi-prover interactive proofs that are sound against non-signaling strategies (nsMIPs). Any nsPCP gives rise to an nsMIP with similar parameters. See [15, 17] for details.

<sup>3</sup> A common relaxation of this condition only requires that the marginals  $\mathcal{F}_{S_1}|_{S_1 \cap S_2}$  and  $\mathcal{F}_{S_2}|_{S_1 \cap S_2}$  are statistically close, rather than equal; a further relaxation only requires these to be computationally close. While we only consider the standard definition (the marginals must equal) we note that [9] shows that this is almost without loss of generality, as every statistically or computationally non-signaling strategy is close to an (exact) non-signaling strategy.

<sup>4</sup> In fact it suffices to only have variables of the form  $X_{S, 1^S}$  since all other probabilities can be computed from these.



Efficiency measures of a nsPCP include familiar notions such as proof length (defined as  $|D|$ ) and the verifier’s randomness and query complexity. In addition, the locality parameter  $k$  controls how hard it is to attain soundness: the smaller  $k$  is, the larger the set of non-signaling functions that the verifier could face. (Note that  $k$ -non-signaling implies  $(k-1)$ -non-signaling.)

There is a qualitative difference between the complexity classes captured by PCPs and by nsPCPs; namely, while PCPs capture *non-deterministic* time languages, nsPCPs capture *deterministic* ones. Indeed, the aforementioned PCP Theorem implies that it is NEXP-hard to approximate the maximum acceptance probability of a PCP verifier (that uses polynomial randomness). In contrast, computing the maximum acceptance probability of an nsPCP verifier that uses  $r$  random bits reduces to a linear program with  $2^{\text{poly}(rk)}$  variables and constraints, a problem solvable in  $\text{EXP} = \cup_{c \in \mathbb{N}} \text{DTIME}(2^{n^c})$ .

If  $k = 2$ , the linear program is solvable in PSPACE [13], which is a tight upper bound [14]. For  $k > 2$  little is known, except for a seminal result of Kalai, Raz, and Rothblum [15, 17], which shows that for  $k = \text{poly}(n)$  it is EXP-hard to approximate a nsPCP verifier’s maximum acceptance probability. In more detail, every language in  $\text{DTIME}(T)$  has a verifier that uses  $\text{poly}(\log T)$  random bits and makes  $\text{poly}(\log T)$  queries to a proof of length  $\text{poly}(T)$ ; soundness holds against  $\text{poly}(\log T)$ -non-signaling functions; the verifier runs in time  $n \cdot \text{poly}(\log T)$  and space  $\text{poly}(\log T)$ .<sup>5</sup>

**The nsPCP Conjecture.** The nsPCP construction behind the above result is a whitebox modification of early PCP constructions [4, 3], and achieves efficiency similar to those. However, modern “PCP technology” goes well beyond these early constructions, via tools such as proof composition [2] and proofs of proximity [10, 6], and enables better efficiency, including the PCP Theorem. Yet, current “nsPCP technology” is limited to the above results, and the question of whether a non-signaling analogue of the PCP Theorem holds remains open.

► **Question 1.** *Is it true that every language in  $\text{DTIME}(T)$  has an nsPCP verifier that uses  $O(\log T)$  random bits, makes  $O(1)$  queries, and is sound against  $O(1)$ -non-signaling functions?*

(As above, we also require that the verifier runs in time  $n \cdot \text{poly}(\log T)$  and space  $\text{poly}(\log T)$ .)

An affirmative answer to the above question would, e.g., improve the hardness result for linear programming in [16], by yielding a reduction that outputs a linear program of polynomial, rather than a quasipolynomial, size. While we do not know if an affirmative answer exists (and we cannot prove that it does not exist), it is clear that the (very few) tools that we have to construct and analyze nsPCPs are far from this goal. In this paper we make headway towards this goal.

## 1.1 Towards a nsPCP Theorem

In [1] a key step towards the PCP Theorem is to prove a weaker result in which the proof has *exponential*, rather than *polynomial*, size (and so the randomness complexity of the verifier is polynomial rather than logarithmic). Namely, one proves that every language in  $\text{NTIME}(T)$  has a PCP verifier that uses  $\text{poly}(T)$  random bits and makes  $O(1)$  queries to a proof of length  $2^{\text{poly}(T)}$ .

<sup>5</sup> Achieving time and space complexities that are  $o(T)$  is important for applications. This is not surprising as every language in  $\text{DTIME}(T)$  has a trivial nsPCP verifier that runs in time  $T$ : the verifier that simply decides the language, without asking any queries. This is unlike the case of PCPs for  $\text{NTIME}(T)$ , where time complexity is less critical.

In this paper we ask whether a non-signaling analogue of this result holds for the class  $\text{DTIME}(T)$ .

► **Question 2.** *Is it true that every language in  $\text{DTIME}(T)$  has an nsPCP verifier that uses  $\text{poly}(T)$  random bits, makes  $O(1)$  queries, and is sound against  $O(1)$ -non-signaling functions?*

We propose this question as a relaxation that, not only is interesting in its own right, but is likely to shed light on Question 1. However, one must be careful with the precise formulation of Question 2. If the verifier can use  $\text{poly}(T)$  random bits then it can simply decide the language by running in time  $T$ , without making any queries. To recover a nontrivial question, we *require* that in order to decide whether an instance  $x$  is in a language  $L \in \text{DTIME}(T)$  the nsPCP verifier first generates queries via a  $\text{poly}(T)$ -time sampler that is *input oblivious* (knows the length of  $x$  but not  $x$  itself), and then rules according to a  $o(T)$ -time decision predicate that knows  $x$ . We stress that all PCP/nsPCP verifiers discussed in this paper are input oblivious.

In this paper we study Question 2 by analyzing a natural candidate construction, and ask:

Is the exponential-length  $O(1)$ -query PCP of [1] sound against  $O(1)$ -non-signaling functions?

Hereafter, we consider the complexity class  $\text{DSIZE}(S)$  (languages decidable by uniform circuits of size  $S(n)$ ) instead of the class  $\text{DTIME}(T)$  (languages decidable by machines in time  $T(n)$ ) because our results, like their classical counterparts, are most easily stated in terms of uniform circuits. This change is only for simplicity, as  $\text{DTIME}(T) \subseteq \text{DSIZE}(\text{poly}(T))$ .

## 1.2 Main theorem

In this paper we prove that the exponential-length constant-query PCP construction of [1] (without modifications) is sound against non-signaling functions. We obtain the following theorem.

► **Theorem 3 (main theorem).** *Every language  $L \in \text{DSIZE}(S)$  has an input-oblivious nsPCP verifier that uses  $O(S^2)$  random bits, makes 11 queries, and is sound against  $O(\log^2 S)$ -non-signaling functions. The query sampler runs in time  $O(S^2)$ , and the decision predicate runs in time  $O(n)$ .*

The theorem is close to answering Question 2, which asks for soundness against  $O(1)$ -non-signaling functions. (See Table 1 for a comparison with the classical result on nondeterministic languages.) At the same time, some may consider Ito’s algorithm [13] as evidence that soundness against  $O(1)$ -non-signaling functions is too much to hope for. Understanding this gap needs further research.

Our result is *incomparable* to the nsPCP of [15, 17], where the nsPCP verifier uses  $\text{poly}(\log S)$  random bits to make  $\text{poly}(\log S)$  queries. The fact that we prove soundness only against  $O(\log^2 S)$ -non-signaling functions (rather than  $O(1)$ -non-signaling functions) is somewhat undesirable, as this implies that the corresponding nsMIP requires  $O(\log^2 S)$  provers. That said, the nsMIP of [15, 17] requires many more provers:  $\text{poly}(\log S)$  with the degree in the polynomial much larger than 2. Another feature of our result is that we have “room” to achieve smaller soundness error *without* using additional provers; for example, by asking more queries to the  $O(\log^2 S)$  provers, we can achieve a sub-constant soundness error of  $2^{-O(\log S)}$ .

Finally, our result is the first to demonstrate that a classical PCP construction is secure against non-signaling functions, *without any modifications*. This should be compared to the construction considered in [15, 17] that, while modeled after the PCP in [4, 3], includes several notable modifications that are needed in the soundness proof.

■ **Table 1** The (linear) ALMSS verifier in different PCP settings.

construction	reference	complexity class	type of PCP	soundness error	proof length	randomness	queries	locality
ALMSS verifier	[1]	NSIZE( $S$ )	PCP	$1 - 1/36$	$2^{O(S^2)}$	$O(S^2)$	11	n/a
+ linearity test	Theorem 3	DSIZE( $S$ )	ns PCP	$1 - 1/10^7$				$O(\log^2 S)$
ALMSS verifier	[1]	NSIZE( $S$ )	LPCP	$3/4$	$O(S^2)$	$O(S)$	4	n/a
	Theorem 4	DSIZE( $S$ )	ns LPCP	$39/40$				$O(\log S)$

### 1.3 Main lemmas

We outline the ideas behind our theorem in Section 2. Concretely, we highlight several statements, which we deem of independent interest, that we prove on the way to the theorem.

Recall that the exponential-length constant-query PCP in [1] is obtained in two steps. First, construct a constant-query verifier where soundness holds as long as the proof string is a *linear function*; this is known as a *linear PCP*. Second, use a linearity test [8] and self-correction to *compile* this linear PCP into a (standard) PCP, where soundness holds against arbitrary proofs.

Our approach follows the same two steps, but adapted to the non-signaling setting. This also departs from the approach in [17], which does not make use of any property testing results.

Note, however, that it is a priori not clear what is the non-signaling analogue of a linear function. A natural attempt would be to say that a non-signaling function  $\mathcal{F}$  is linear iff it passes the BLR linearity test with probability 1 (where the probability is over the test and  $\mathcal{F}$ ). But this attempt is awkward, because the definition depends on a local test, and avoids discussing “global” structure.

A recent work [9] tells us that the right definition is to say that  $\mathcal{F}$  is linear iff it corresponds to a *quasi-distribution* over linear functions. A quasi-distribution is a probability distribution where the weights can be any real number and are not restricted to be in  $[0, 1]$ . Quasi-distributions over functions arise in this context because they are an equivalent description of non-signaling functions.

In light of the above, the notion of a *non-signaling linear PCP* (nsLPCP) is immediate: the definition requires soundness to hold against all *linear* non-signaling functions.

The first step in our proof is showing that the linear PCP verifier of [1] (the “ALMSS verifier”), when used for deterministic computations, is sound against linear non-signaling functions.

► **Theorem 4.** *The (input oblivious) ALMSS verifier, for a given language  $L \in \text{DSIZE}(S)$ , uses  $O(S)$  random bits, makes 4 queries, and is sound against linear  $O(\log S)$ -non-signaling functions.*

See Table 1 for a comparison with the classical result showing soundness against linear functions.

In order to “lift” Theorem 4 to Theorem 3, we need a suitable linearity test.

The linearity test of [8] was recently analyzed in the non-signaling setting by [9], who proved that any  $k$ -non-signaling function  $\mathcal{F}$  that passes the linearity test with probability  $1 - \varepsilon$  can be self-corrected to a  $\lfloor k/2 \rfloor$ -non-signaling function  $\hat{\mathcal{F}}$  that is  $2^{O(k)}\varepsilon$ -close to a linear  $\lfloor k/2 \rfloor$ -non-signaling function  $\mathcal{L}$ . (Self-correction and closeness have precise meanings, discussed later.) However, we cannot directly use [9]’s result, because in our theorem the locality parameter  $k$  is required to be super-constant ( $k = O(\log S)$  in Theorem 4), and thus

the bound on the distance between  $\hat{\mathcal{F}}$  and  $\mathcal{L}$  is too large, even when considering only query sets of small size. Specifically, we need the distance to be a sufficiently small constant on query sets of size 4 (the number of queries in Theorem 4).

We solve this problem by extending the result in [9] in a black-box way and proving that the distance between  $\hat{\mathcal{F}}$  and  $\mathcal{L}$  on a query set  $Q$  is only  $O(|Q| \sqrt{\varepsilon})$ , provided that the error  $\varepsilon$  and  $\mathcal{L}$ 's locality are sufficiently small. Crucially, if  $|Q|$  is constant, so is the distance between  $\hat{\mathcal{F}}$  and  $\mathcal{L}$ . The proof of this statement involves analyzing the *repeated* linearity test, whose behavior in the non-signaling setting is quite subtle when compared to the classical setting (see Section 2.6).

► **Theorem 5.** *Let  $k, \bar{k} \in \mathbb{N}$  and  $\varepsilon \in (0, 1/400]$  be such that  $k = \Omega((\bar{k} + \log \frac{1}{\varepsilon}) \cdot \bar{k})$ . If a  $k$ -non-signaling function  $\mathcal{F}: \{0, 1\}^n \rightarrow \{0, 1\}$  passes the linearity test with probability at least  $1 - \varepsilon$  then there exists a linear  $\bar{k}$ -non-signaling function  $\mathcal{L}: \{0, 1\}^n \rightarrow \{0, 1\}$  such that for all query sets  $Q \subseteq \{0, 1\}^n$  with size  $|Q| \leq \bar{k}$  and for all events  $E \subseteq \{0, 1\}^Q$  it holds that*

$$\left| \Pr[\hat{\mathcal{F}}(Q) \in E] - \Pr[\mathcal{L}(Q) \in E] \right| \leq O(|Q| \sqrt{\varepsilon}) .$$

The above result on linearity testing enables us to transform our nsLPCP, and more generally *any* nsLPCP, into a corresponding nsPCP with minimal changes in parameters (the transformation is exactly the classical compiler). This is the last key statement in the proof of our main theorem.

► **Lemma 6.** *For every  $\varepsilon \in [0, 1]$ , if a language  $L$  has an nsLPCP where the verifier uses  $r$  random bits, makes  $q$  queries, and has soundness error  $1 - \varepsilon$  against linear  $k$ -non-signaling functions  $\mathcal{L}: \{0, 1\}^\ell \rightarrow \{0, 1\}$ , then  $L$  has an nsPCP where the verifier uses  $r + O(q\ell)$  random bits, makes  $O(q)$  queries, and has soundness error  $1 - O_q(\varepsilon^2)$  against  $O_\varepsilon(k^2)$ -non-signaling functions  $\mathcal{F}: \{0, 1\}^\ell \rightarrow \{0, 1\}$ . (Furthermore, if the former is input oblivious, so is the latter.)*

## 1.4 Enriching the toolkit for non-signaling PCPs

Progress in our understanding of PCPs has typically moved hand in hand with progress in our understanding of low-degree testing. In particular, many PCP constructions follow this blueprint:

- (1) a low-degree test that, via only a few queries, ensures that a given proof conforms to a specified algebraic encoding;
- (2) a probabilistic test that, assuming the proof is (essentially) given in this encoding, ensures that the statement being checked is true with high probability.

In contrast, while the nsPCP in [17] is reminiscent of this blueprint, its analysis does not follow it, despite the fact that the construction is modeled after the PCP in [4, 3], for which the two-step analysis *is* possible (in the classical setting). The lack of such general paradigms means that we lack general design principles to construct better nsPCPs.

This state of affairs raises the intriguing question of whether a theory of low-degree testing (and, more generally, property testing) is feasible in the non-signaling setting and, moreover, whether one can build on it to construct nsPCPs in order to make further progress towards Question 1.

An additional contribution of our work is to *enrich* the current “non-signaling toolkit”, by demonstrating an example where the aforementioned blueprint is both possible and useful.

Namely, building on the work of [9] on linearity testing, our results provide a modular paradigm that not only simplifies the overall analysis, thereby enabling us to assert that the construction of [1] *with no modifications* is sound against non-signaling strategies, but also (as discussed later) clarifies the technical barriers that separate us from answering Question 2. All this suggests that our techniques will be helpful for constructing more efficient nsPCPs.

## 1.5 Concurrent work

In a concurrent work, Kiyoshima [20] studies the soundness of ALMSS-type PCPs against non-signaling strategies. Kiyoshima proves that, for a sufficiently large security parameter  $t$  (at least logarithmic in the circuit size), the  $t$ -repetition of a  $O(t)$ -query modification of the ALMSS-verifier has soundness error  $\text{negl}(t)$  against  $O(t^2)$ -non-signaling functions. In comparison, we prove that the *unmodified* 11-query ALMSS PCP has soundness error  $O(1)$  against  $O(\log^2 S)$ -non-signaling strategies (and also that a modification of its  $t$ -repetition has soundness error  $\exp(-t)$  for every  $t = \Omega(\log S)$ ). While both our analysis and Kiyoshima's analysis avoid the use of an augmented circuit (necessarily so as it would have had exponential size), our techniques differ. Kiyoshima conducts a direct analysis of the PCP verifier, while we adopt a modular approach in which we first prove soundness against *linear* non-signaling strategies (a simpler task), and then, building on a recent analysis of the linearity test [9], we deduce soundness against *all* non-signaling strategies. We consider the modular and simple analysis in our work to be of independent interest. Kiyoshima additionally proves that soundness holds against *computational* non-signaling strategies, a relaxation where the marginal distributions on intersections are only required to be computationally close. Our results directly extend to computational non-signaling strategies as every computational non-signaling strategy is close to an exact non-signaling strategy (as proved in [9]).

In another concurrent work, Holmgren and Rothblum [12] study the problem of constructing PCPs/MIPs in which the prover is very efficient in time and space [7], in the non-signaling setting. While they consider a construction that is more closely related to the PCP in [4, 3] (honest proofs are encoded via low-degree polynomials rather than linear functions), their soundness analysis also has the feature that it avoids the use of an augmented circuit.

## 1.6 Open problems

The question of whether the exponential-length constant-query PCP of [1] is sound against  $O(1)$ -non-signaling functions remains open. A concrete approach to affirmatively answer this question is to prove that the *linear* PCP verifier of [1] is sound against  $k$ -non-signaling functions for  $k = O(1)$ , rather than  $k = O(\log S)$  as in Theorem 4. (Our generic compiler from Theorem 6 would then take care of the rest.) Another intriguing possibility is that an affirmative answer to Question 2 could come from a *different* exponential-size constant-query PCP. However, the result due to [13] shows that the class of nsMIPs with 2 provers equals PSPACE, which possibly suggests that soundness against  $O(1)$ -non-signaling functions is too much to hope for.

Moreover, while our results can be interpreted as progress towards a non-signaling analogue of the PCP Theorem (Question 1), it remains unclear whether such an analogue holds, and more investigations in nsPCPs are needed. We believe that our work and our new techniques can inform such investigations.

## 2 Techniques

We outline the techniques used to prove our results. First, in Section 2.1, we explain the transformation from a nsLPCP to a corresponding nsPCP. Next, in Sections 2.2 to 2.5 we discuss the nsLPCP on which we apply this transformation, namely, the ALMSS verifier [1]. Finally, in Section 2.6, we discuss linearity testing with low error, which underlies the transformation.

## 2.1 From nsLPCP to nsPCP

We discuss the transformation from nsLPCP to nsPCP (Theorem 6). We first recall the classical transformation from LPCP to PCP, and then explain how to achieve its non-signaling analogue.

**The classical case.** The classical transformation from LPCP to PCP relies on the following tools.

- *Testing linearity.* Given a boolean function  $f: \{0, 1\}^\ell \rightarrow \{0, 1\}$ , the linearity test draws random  $x, y \in \{0, 1\}^\ell$  and checks that  $f(x) + f(y) = f(x + y)$  [8]. If the test passes with probability  $1 - \varepsilon$ , then  $f$  is  $\varepsilon$ -close to a linear function  $f^*: \{0, 1\}^\ell \rightarrow \{0, 1\}$  [8, 5].
- *Self-correction.* Given  $f$  that is  $\varepsilon$ -close to a linear function  $f^*$ , one can create a probabilistic oracle  $\mathcal{O}$  that, given any  $x \in \{0, 1\}^\ell$ , returns  $f^*(x)$  with probability  $1 - 2\varepsilon$ . Namely,  $\mathcal{O}$  samples a random  $z \in \{0, 1\}^\ell$ , queries  $f$  on  $z + x$  and  $z$ , and answers with  $f(z + x) - f(z)$ .

The above tools imply a transformation from LPCP to PCP: given access to an arbitrary function  $f: \{0, 1\}^\ell \rightarrow \{0, 1\}$ , the PCP verifier runs the linearity test and then runs the LPCP verifier by self-correcting each of its queries. If the LPCP verifier makes  $q$  queries and has soundness error  $\gamma$ , then the resulting PCP verifier makes  $3 + 2q$  queries and has soundness error  $\max\{1 - \varepsilon, \gamma + 2q\varepsilon\}$ , where  $\varepsilon$  is (a bound on) the distance of  $f$  to linear functions. This soundness error is bounded by  $1 - \frac{1-\gamma}{2q+1}$  (the maximum is when the two terms equal), which is bounded away from 1.

If desired, the soundness error can be made arbitrarily close to  $\gamma$  by repeating the linearity test. Given a parameter  $t$ , the repeated linearity test samples  $x_i, y_i \in \{0, 1\}^\ell$  for each  $i \in [t]$  and checks that  $f(x_i) + f(y_i) = f(x_i + y_i)$  for all  $i \in [t]$ . Now, the PCP verifier makes  $3t + 2q$  queries and has soundness error  $\max\{(1 - \varepsilon)^t, \gamma + 2q\varepsilon\}$ , which for suitable  $\varepsilon$  and  $t = O_{\gamma, \varepsilon}(q)$  is arbitrarily close to  $\gamma$ .

**The non-signaling case.** We follow the structure of the classical transformation. However, the non-signaling case not only calls for a different analysis but also raises a problem that we must solve.

The linearity test in the non-signaling setting has the following guarantee [9]: if  $\mathcal{F}$  is a  $k$ -non-signaling function such that  $\Pr_{x, y, \mathcal{F}}[\mathcal{F}(x) + \mathcal{F}(y) = \mathcal{F}(x + y)] \geq 1 - \varepsilon$  then  $\mathcal{F}$  can be self-corrected (in the natural way) to a  $(k/2)$ -non-signaling function  $\hat{\mathcal{F}}$  that is  $2^{O(k)}\varepsilon$ -close to a linear non-signaling function  $\mathcal{L}$ . Note that self-correction is already part of the conclusion.

The above result appears sufficient for compiling a nsLPCP verifier into a corresponding nsPCP verifier. Namely, given a  $k$ -non-signaling function  $\mathcal{F}: \{0, 1\}^\ell \rightarrow \{0, 1\}$ , the nsPCP verifier checks that  $\mathcal{F}(x) + \mathcal{F}(y) = \mathcal{F}(x + y)$  for random  $x, y \in \{0, 1\}^\ell$  and also checks that the nsLPCP verifier accepts  $\hat{\mathcal{F}}$ . Analogously to before, if the nsLPCP verifier makes  $q$  queries and has soundness error  $\gamma$  against linear  $(\frac{k-3}{2})$ -non-signaling functions, then the resulting PCP verifier makes  $3 + 2q$  queries and has soundness error  $\max\{1 - \varepsilon, \gamma + 2^{O(k)}\varepsilon\}$  against arbitrary  $k$ -non-signaling functions.

However, our analysis of the ALMSS verifier (the nsLPCP that we use) will require locality  $k = \Omega(\log N)$ , which means that the additive term  $2^{O(k)}\varepsilon$  grows with  $N$ . This precludes achieving a constant soundness error with constant query complexity.

The foregoing motivates the problem of testing linearity of non-signaling functions *with low error*: how do we ensure that  $\hat{\mathcal{F}}$  is sufficiently close to a linear non-signaling function  $\mathcal{L}$ ? We stress that while in the classical case improving the “quality” of the self-correction has a straightforward solution (repeat the linearity test, and do self-correction), in the non-signaling case this problem is quite involved. Moreover, *we do not wish to modify in any way the classical compiler*, and thus relying on additional queries (even if only a constant number depending on  $q$  and  $\varepsilon$ ) is not an option.



We discuss our solution to this problem later on in Section 2.6, thereby providing the missing ingredient of our compiler from nsLPCP to nsPCP. In the meantime, in Sections 2.2 to 2.5, we discuss how we prove that the ALMSS verifier is secure against linear non-signaling functions.

## 2.2 The linear ALMSS verifier against linear non-signaling functions

Our goal is to establish that the linear PCP verifier of [1] (the “ALMSS verifier”) is sound against linear non-signaling functions, and thus prove that every language  $L \in \text{DSIZE}(S)$  has a constant-query nsLPCP verifier that is sound against linear  $O(\log S)$ -non-signaling functions. Note that we invoke the ALMSS verifier on *deterministic* (DSIZE) computations, rather than on *nondeterministic* (NSIZE) computations as in the classical case. We now recall the ALMSS verifier.

Let  $L \in \text{DSIZE}(S)$  be a language, and let  $\{C_n\}_{n \in \mathbb{N}}$  be a uniform boolean circuit family of size  $N := S(n)$  that decides  $L$  (for all  $x \in \{0, 1\}^n$ ,  $x \in L$  iff  $C_n(x) = 1$ ). Hereafter we omit the subscript in  $C_n$  as it is clear from context. Given an input  $x$ , one can express the condition “ $C(x) = 1$ ” as a system of simple equations over  $C$ ’s wires  $W$ ; the variables are  $\mathbf{w} = (w_1, \dots, w_N)$ , one per wire. We use the convention that the input wires are  $w_1, \dots, w_n$  and the output wire is  $w_N$ . To ensure input consistency we need that  $w_j = x_j$  for every  $j \in \{1, \dots, n\}$ ; to ensure correct gate computations we need that, for every  $j \in \{n+1, \dots, N\}$ ,  $w_j$  is the correct combination of the variables used to compute it (e.g., if  $w_j$  is the output of an AND gate with inputs  $w_{j_1}$  and  $w_{j_2}$  then the equation is  $w_j = w_{j_1} \cdot w_{j_2}$ ); to ensure that the output is 1 we need that  $w_N = 1$ . This can be summarized as a system of  $M := N + 1$  equations  $\{P_j(\mathbf{w}) = c_j\}_{j \in [M]}$ , where  $P_1, \dots, P_M$  are quadratic polynomials (each involving at most three variables in  $\mathbf{w}$ ) and  $c_1, \dots, c_M$  are boolean constants.

The ALMSS verifier is given below. We overload notation and use  $P_j$  to also denote the upper triangular matrix in  $\{0, 1\}^{N^2}$  such that  $P_j(\mathbf{w}) = \langle P_j, \mathbf{w} \otimes \mathbf{w} \rangle$ ; that is, if  $P_j(\mathbf{w}) = \sum_{i=1}^N a_i w_i + \sum_{1 \leq i < i' \leq N} a_{i,i'} w_i w_{i'}$ , then  $P_j$  has  $a_i$  in the diagonal entry  $(i, i)$  and  $a_{i,i'}$  in the entry  $(i, i')$ , for  $1 \leq i < i' \leq N$ . Also, for  $a \in \{0, 1\}^N$ ,  $D_a$  is the diagonal matrix in  $\{0, 1\}^{N^2}$  whose diagonal is  $a$ .

The ALMSS verifier, given input  $x \in \{0, 1\}^n$  and oracle access to a linear non-signaling function  $\mathcal{L}: \{0, 1\}^{N^2} \rightarrow \{0, 1\}$ , works as follows:

1. Use the circuit  $C$  and input  $x$  to construct the matrices  $P_1, \dots, P_M \in \{0, 1\}^{N^2}$  and constants  $c_1, \dots, c_M \in \{0, 1\}$ , which represent the computation of  $C$  on  $x$ .
2. Draw random  $s \in \{0, 1\}^M$ ,  $u, v \in \{0, 1\}^N$ , and query  $\mathcal{L}$  on the set  $\{\sum_{j=1}^M s_j P_j, D_u, D_v, u \otimes v\}$ .
3. Check that  $\mathcal{L}(\sum_{j=1}^M s_j P_j) = \sum_{j=1}^M s_j c_j$  and that  $\mathcal{L}(D_u)\mathcal{L}(D_v) = \mathcal{L}(u \otimes v)$ .

If  $C(x) = 1$ , the honest proof is the linear function  $\pi: \{0, 1\}^{N^2} \rightarrow \{0, 1\}$  where  $\pi(Z) := \langle \mathbf{w} \otimes \mathbf{w}, Z \rangle = \sum_{i,i' \in [N]} w_i w_{i'} \cdot Z_{i,i'}$  where  $w_i$  is now the value of the  $i$ -th wire in the computation of  $C$  on  $x$ .

The challenge is to prove that the ALMSS verifier is sound against *linear non-signaling functions*. Namely, we must show that if there is a linear non-signaling function  $\mathcal{L}$  that is accepted with good probability then  $x \in L$ , or equivalently that  $C(x) = 1$ . We discuss this in the next sub-sections.

### 2.3 A linear local assignment generator suffices

The first step in our soundness analysis shows that, to establish that  $C(x) = 1$ , it suffices to construct a *linear local assignment generator* with sufficiently small error.

A linear  $k$ -local assignment generator for  $(C, x)$  with error  $\varepsilon$  is a linear  $k$ -non-signaling function  $\mathcal{A}: \{0, 1\}^N \rightarrow \{0, 1\}$  that individually satisfies each of the  $M$  constraints with probability  $1 - \varepsilon$  (over the randomness of  $\mathcal{A}$ ). Namely,

- (a) for each  $i \in \{1, \dots, n\}$ ,  $\Pr[\mathcal{A}(e_i) = x_i] \geq 1 - \varepsilon$ ;
- (b) for each  $i \in \{n + 1, \dots, N\}$ , if  $w_i$  is the output of a unary gate  $g$  with input  $w_j$  then  $\Pr[\mathcal{A}(e_i) = g(\mathcal{A}(e_j))] \geq 1 - \varepsilon$ , else if  $w_i$  is the output of a binary gate  $g$  with inputs  $w_{j_1}, w_{j_2}$  then  $\Pr[\mathcal{A}(e_i) = g(\mathcal{A}(e_{j_1}), \mathcal{A}(e_{j_2}))] \geq 1 - \varepsilon$ ;
- (c)  $\Pr[\mathcal{A}(e_N) = 1] \geq 1 - \varepsilon$ .

(Here  $e_i$  is the  $i$ -th vector in the standard basis.)

► **Lemma 7 (informal).** *If there exists a  $k$ -local assignment generator for  $(C, x)$  with error  $\varepsilon$  for  $k = \Omega(\log N)$  and  $\varepsilon = O(\frac{1}{N \log N})$ , then  $C(x) = 1$ .*

We sketch the proof of this lemma. The transcript of the computation of  $C$  on  $x$  is the *unique* correct assignment to all the wires. We say that a wire  $w_i \in W$  of  $C$  is *correct* whenever  $\mathcal{A}(e_i)$  equals the value contained in this transcript; more generally, we say that a vector  $z \in \{0, 1\}^N$  is correct if  $\mathcal{A}(z)$  equals the value of  $z$  in the linear extension of the transcript. Below, we partition  $C$ 's wires  $W$  into layers  $W_1, \dots, W_H$  according to depth. (We assume layered circuits.)

As a warmup, suppose for now that  $k \geq N$ . The probability that all wires in  $W_1$  are correct is at least  $1 - |W_1|\varepsilon$ , and the probability that all the gates are correct is at least  $1 - \sum_{h=2}^H |W_h|\varepsilon$ . Therefore by union bound, the probability that all wires in the circuit are correct is at least  $1 - \sum_{h=1}^H |W_h|\varepsilon$ , because if all the input wires are correct and all the gates are computed correctly, then all the wires in the circuit are correct. In particular, we deduce that the output wire is correct with probability  $1 - \sum_{h=1}^H |W_h|\varepsilon = 1 - |W|\varepsilon = 1 - N\varepsilon$ . Since the output wire is 1 with probability  $1 - \varepsilon$ , and  $\varepsilon = O(\frac{1}{N})$ , we conclude that  $\Pr[C(x) = 1] > 0$ , and thus  $C(x) = 1$ .

The above argument requires that  $k \geq N$ , because we have to simultaneously “view” assignments to all wires in the circuit. While the argument can be easily modified so that we only require  $k$  to be at least twice the width of  $C$ , the latter may still be much larger than  $O(\log N)$ .

Using the linearity of  $\mathcal{A}$ , however, we can modify the argument to merely require  $k = \Omega(\log N)$ . For each layer  $h$ , we define an event  $E_h$  such that if  $E_h$  holds, then any wire in layer  $h$  is correct with high probability. In the warmup above  $E_h$  is the event “all wires in layer  $h - 1$  are correct”; in our proof  $E_h$  is the event “ $t$  random linear combinations of wires in layer  $h$  are correct”. Given a wire  $w_i$  in layer  $h$ , we can bound the event “ $\mathcal{A}(e_i)$  is incorrect and  $E_h$  holds” as follows. If  $\mathcal{A}(e_i)$  is incorrect, then all linear combinations of wires in layer  $h$  can be split into pairs  $z$  and  $z + e_i$ , and exactly one of  $\mathcal{A}(z)$  and  $\mathcal{A}(z + e_i)$  is incorrect. Hence, the probability that a random linear combination of wires in layer  $h$  is correct, given that  $\mathcal{A}(e_i)$  is incorrect, is at most  $1/2$ , and so  $\Pr[E_h \mid \mathcal{A}(e_i) \text{ is incorrect}] \leq 2^{-t}$ , since the  $t$  random linear combinations are independent. Using Bayes’s rule (and an additional assumption that  $\Pr[E_h] \geq 1/2$ ), we deduce that  $\Pr[\mathcal{A}(e_i) \text{ is incorrect} \mid E_h]$  is small. We then proceed inductively on the layers as before.

The argument above requires that  $\varepsilon = O(\frac{1}{N \log N})$ . One may wonder whether a similar result could be proved with, say,  $\varepsilon = O(1)$ . We additionally prove that our analysis is almost tight, in that an error of  $\varepsilon = O(\frac{\log N}{N})$  is necessary, *regardless* of how large the locality  $k$  is.

See the full version of this paper for details.



**Local assignment generators in prior works.** Local assignment generators appear in prior works on nsPCPs [17, 22], but our notion is qualitatively different, as we now explain.

Prior works consider local assignment generators for an *augmented* circuit  $C_{\text{aug}}$  rather than for  $C$  itself. (See Section 1.5 for a discussion of concurrent work that avoids augmented circuits.) Informally,  $C_{\text{aug}}$  not only contains  $C$  as a sub-circuit but also low-degree extensions of  $C$ 's layers as well as subcircuits computing all low-degree tests on these. The wires contained in these additional subcircuits are what enables defining an event  $E_h$  on which to condition for each layer.

The analogue of the augmented circuit  $C_{\text{aug}}$  in our setting, however, has *exponential* size, and thus *we cannot use it*. Namely, we would have to encode each layer of  $C$  via the Hadamard code (all linear combinations of wires in the layer) and then compute all possible linear tests on these.

Instead, our assumption that the local assignment generator is a *linear* non-signaling function implies that we *do not have to construct an augmented circuit*. Namely, the linear combinations that we use to define the event  $E_h$  are implicitly available due this linearity, and so there is no need to augment  $C$  (nor, in particular, to introduce any gates that evaluate linearity tests).

The assumption that the local assignment generator is linear is justified by the fact that a different part of our construction (the linearity test in our generic compiler) ensures the non-signaling function is (close to) linear. Overall, this separation not only avoids the aforementioned issues of using augmented circuits, but also simplifies the analysis of the local assignment generator.

## 2.4 Constructing the linear local assignment generator

Given a  $k$ -non-signaling function  $\mathcal{L}: \{0, 1\}^{N^2} \rightarrow \{0, 1\}$  that is accepted by the ALMSS verifier with probability at least  $1 - \varepsilon$ , we can obtain a linear  $k$ -local assignment generator  $\mathcal{A}: \{0, 1\}^N \rightarrow \{0, 1\}$  with error  $O(\varepsilon)$  by “restricting  $\mathcal{L}$  to its diagonal”. Namely, in order to query  $\mathcal{A}$  at  $v \in \{0, 1\}^N$ , we query  $\mathcal{L}$  at  $D_v \in \{0, 1\}^{N^2}$ , where  $D_v$  is the diagonal matrix that has  $v$  as its diagonal.

We show that, since  $\mathcal{L}$  is accepted with probability at least  $1 - \varepsilon$ ,  $\mathcal{L}$  must satisfy any *individual* constraint  $P_j(\mathbf{w}) = c_j$  with probability at least  $1 - O(\varepsilon)$ , and this directly implies that the linear local assignment generator  $\mathcal{A}$  has error  $O(\varepsilon)$ . (See the full version of this paper for details.)

The discussion so far already gives us a weak bound on the soundness error of the ALMSS verifier, namely  $1 - O(\frac{1}{N \log N})$ . Indeed, for  $k = O(\log N)$  and  $\varepsilon = O(\frac{1}{N \log N})$ , we can apply the lemma above (in Section 2.3) to conclude that  $C(x) = 1$ .

However, our goal is to show that the ALMSS verifier (as is) has *constant* soundness error, and doing so requires more technical work, which we discuss next.

► **Remark.** We stress that proving a soundness error of even  $1 - O(\frac{1}{N \log N})$  is a non-trivial statement. This is in contrast to the classical setting, where if an assignment satisfies an  $1 - \varepsilon$  fraction of the  $M = N + 1$  constraints for  $\varepsilon < 1/M$ , then, trivially, *all* constraints are satisfied.

## 2.5 The ALMSS verifier has constant soundness error

Our goal is to prove that the ALMSS verifier has constant soundness error. In a first step (Section 2.5.1), we use the soundness error proved above (Section 2.4) to show that the  $t$ -repeated ALMSS verifier has soundness error  $\gamma$  when  $t = \Omega(\log N + \log \frac{1}{\gamma})$ . In a second

step (Section 2.5.2), we prove that the basic ALMSS verifier (no repetitions) has constant soundness error. The second step is *generic* and of independent interest: we prove that if a  $t$ -repeated verifier has soundness error  $\exp(-t)$ , then the corresponding basic verifier has soundness error  $O(1)$ .

### 2.5.1 The $t$ -repeated ALMSS verifier has soundness error $\exp(-t)$

While in the classical setting reducing soundness error via simple repetition is straightforward ( $t$ -wise repetition reduces soundness error from  $\delta$  to  $\delta^t$ ), in the non-signaling setting simple repetition *does not work*.<sup>6</sup> Indeed, consider the non-signaling function (in fact, distribution) that, with probability  $1 - \varepsilon$ , answers the verifier's queries in an accepting way, and otherwise answers randomly. This non-signaling function is accepted by the  $t$ -repeated verifier with probability  $\approx 1 - \varepsilon$ , which is about the same as the probability that it is accepted by a single verifier.

However, this example provides intuition for how one circumvents this issue. Informally, we would like to extract the “ $1 - \varepsilon$  good part” that satisfies the verifier, and drop the “ $\varepsilon$  bad part”. We follow a technique used in [17] and, instead of arguing about the probability that  $\mathcal{L}$  passes the  $t$ -repeated verifier, we argue that the non-signaling function  $\mathcal{L}$  *conditioned on passing the  $t$ -repeated verifier* passes the basic verifier with high probability. Indeed, in the aforementioned example, conditioning on at least one test passing removes the “ $\varepsilon$  bad part” injected by the distribution, and intuitively *extracts* the part of  $\mathcal{L}$  that is passing the verifier. An interesting feature of our analysis of the verifier is that our conclusion is about the basic verifier, not the relaxed  $t$ -repeated verifier, which plays a major role in the analysis in [17].<sup>7</sup> This is a qualitative difference in our analysis arising from our use of property testing (not present in [17]), which also simplifies the analysis.

In more detail, let  $\mathcal{L}'$  denote the linear non-signaling function that equals  $\mathcal{L}$  when conditioned on passing the  $t$ -repeated verifier. Namely, if  $E$  is the (random) event that  $\mathcal{L}$  passes the  $t$ -repeated verifier, then for any  $S \subseteq \{0, 1\}^n$  (of some maximal size) and  $\vec{b} \in \{0, 1\}^S$ , we define

$$\Pr[\mathcal{L}'(S) = \vec{b}] := \Pr[\mathcal{L}(S) = \vec{b} \mid E] = \frac{\Pr[\mathcal{L}(S) = \vec{b} \wedge E]}{\Pr[E]}. \quad (1)$$

We then prove that  $\mathcal{L}'$  passes the basic verifier with probability at least  $1 - \frac{1/\Pr[E]}{\exp(t)}$ .

The proof uses a generic lemma stating that, if we run  $t + d$  independent tests, then the probability that at most  $r$  out of the first  $d$  tests pass and all of the last  $t$  tests pass is at most  $(\frac{d}{t+d})^{r+1}$ . A naive application of this lemma (with  $r = 0$  and  $d = 1$ ) shows that  $\mathcal{L}'$  passes the basic verifier with probability at least  $1 - \frac{1/\Pr[E]}{(t+1)}$ . This is not enough, because (using  $\Pr[E] \geq \gamma$ ) we would require  $t = \Omega(N \log N \cdot \frac{1}{\gamma})$  to prove soundness, which is again far too many repetitions.

However, we leverage the linearity of  $\mathcal{L}$  to deduce the stronger guarantee, as we now explain. We want to bound the probability that  $\mathcal{L}'$  does not pass the basic verifier, which means we need to bound the probability that  $\mathcal{L}$  fails exactly the first test of  $t + 1$  independent tests. We do this by arguing this individually for each of the two types of tests made by

<sup>6</sup> Even if simple repetition were to reduce soundness error from  $\delta$  to  $\delta^t$ , then to get  $\delta^t = \gamma$  when  $\delta = 1 - O(\frac{1}{N \log N})$  we would need to repeat  $t = \Omega(N \log N + \log \frac{1}{\gamma})$  times, which requires too large of a locality  $k$  for the analysis.

<sup>7</sup> The relaxed  $t$ -repeated verifier runs  $t$  tests and accepts if a large fraction of them pass.

the ALMSS verifier: the tensor test “ $\mathcal{L}(D_u)\mathcal{L}(D_v) = \mathcal{L}(u \otimes v)$ ” and the satisfiability test “ $\mathcal{L}(\sum_{j=1}^M s_j P_j) = \sum_{j=1}^M s_j c_j$ ”. We will explain our techniques in the case of the satisfiability test; the same techniques work for the tensor test, but the algebra is messier.

In the case of the satisfiability test, we split the “special” test (i.e., the first one) into  $d$  pairs of tests, such that each individual test is random, but each pair is correlated so that if both tests in some pair pass, then the original test passes. Specifically, we draw  $d$  random vectors  $s^{(1)}, \dots, s^{(d)} \in \{0, 1\}^M$ , and then we split the test “ $\mathcal{L}(\sum_{j=1}^M s_j P_j) = \sum_{j=1}^M s_j c_j$ ” into the  $d$  pairs of tests

$$\mathcal{L}\left(\sum_{j=1}^M (s_j + s_j^{(i)}) P_j\right) = \sum_{j=1}^M (s_j + s_j^{(i)}) c_j \quad \text{and} \quad \mathcal{L}\left(\sum_{j=1}^M s_j^{(i)} P_j\right) = \sum_{j=1}^M s_j^{(i)} c_j .$$

This allows us to apply the lemma with  $d = O(t)$ , and  $r = O(t)$ , which shows that  $\mathcal{L}'$  passes the basic verifier with probability at least  $1 - \frac{1/\Pr[E]}{\exp(t)}$ , an exponential decrease in  $t$ .

The above analysis shows soundness error of  $\gamma$  for the  $t$ -repeated verifier, for  $t = O(\log N + \log \frac{1}{\gamma})$ . Indeed, by the above argument, the conditioned function  $\mathcal{L}'$  passes the basic verifier with probability  $1 - \frac{1}{\gamma} \exp(-t) = 1 - O(\frac{1}{N \log N})$ , by choice of  $t$ . The analysis in the previous section (Section 2.4) then implies that  $C(x) = 1$ , proving soundness of the  $t$ -repeated verifier.

Setting  $\gamma = \exp(-t)$ , the discussion so far merely shows that the  $t$ -wise repetition of the ALMSS verifier, which makes  $4t$  queries, has soundness error  $\exp(-t)$  when  $t = \Omega(\log N)$ ; moreover, we get no conclusions for  $t = o(\log N)$ . But we still did not conclude anything about the soundness of a *single* invocation of the 4-query ALMSS verifier. We next discuss how to handle this case.

## 2.5.2 Back to the 4-query ALMSS verifier

We establish that the ALMSS verifier has constant soundness error by proving a generic lemma. The lemma states that, for *any* PCP verifier  $V$ , if the  $t$ -repeated verifier  $V^t$  has soundness error  $\exp(-t)$ , then  $V$  has soundness error  $O(1)$ . Since we have already argued that the  $t$ -repeated ALMSS verifier has soundness error  $\exp(-t)$  for  $t = \Omega(\log N)$ , we can conclude that the basic ALMSS verifier has soundness error  $O(1)$ , against  $O(\log N)$ -non-signaling linear functions.

In the classical case, the proof of this generic fact is trivial: a (classical) function passes a PCP verifier  $V$  with probability  $\delta$  if and only if it passes the  $t$ -repeated verifier  $V^t$  with probability  $\delta^t$ . However, in the non-signaling case, it is not clear what one can say because a non-signaling function can provide correlated answers across repetitions. Nevertheless, we are able to *lower bound* the probability that  $V^t$  accepts by a quantity that is almost  $\delta^t$  (which is, in particular, almost tight).

To our knowledge, we are the first to relate the soundness of  $V$  to the soundness of  $V^t$ . Generic statements in prior works (starting with [15]) have related the soundness of  $V^t$  to the soundness of the  $t$ -repeated relaxed verifier (which accepts if a vast majority of the  $t$  tests pass), but did not provide conclusions about the basic verifier  $V$ .

## 2.6 Testing linearity with low error

Below we discuss linearity testing *with low error* (Theorem 5) in more detail.

**Warmup: distributions.** We have discussed (in Section 2.1) how to test linearity with low error in the classical setting. In order to illustrate some of the difficulties that arise in the non-signaling setting, we first discuss a special case of it: testing linearity against a *distribution* over functions.

First, suppose that  $\mathcal{D}$  is a distribution over functions  $f: \{0,1\}^n \rightarrow \{0,1\}$  that passes the linearity test with probability  $1 - \varepsilon$ . The self-correction  $\hat{\mathcal{D}}$  that on input  $x \in \{0,1\}^n$  samples a random  $z \in \{0,1\}^n$  and outputs  $\hat{\mathcal{D}}(x) = \mathcal{D}(z+x) - \mathcal{D}(z)$  is  $2\varepsilon$ -close to a distribution over *linear* functions  $\mathcal{D}^*$ , namely, for every  $x \in \{0,1\}^n$  it holds that  $|\Pr[\hat{\mathcal{D}}(x) = 1] - \Pr[\mathcal{D}^*(x) = 1]| \leq 2\varepsilon$ . Indeed, consider the distribution  $\mathcal{D}^*$  that samples  $f \leftarrow \mathcal{D}$  and outputs any linear function  $f^*$  closest to  $f$ .<sup>8</sup> Then, for every function  $f$  and  $x \in \{0,1\}^n$ , the probability over a random  $z \in \{0,1\}^n$  that  $f^*(z) = f(z)$  and  $f^*(z+x) = f(z+x)$  is at least  $1 - 2\varepsilon_f$ , where  $\varepsilon_f := 1 - \Pr_{x,y}[f(x) + f(y) = f(x+y)]$ . Denoting by  $d_f$  denotes the probability that  $\mathcal{D}$  samples the function  $f$ , we conclude that  $|\Pr[\mathcal{D}^*(x) = 1] - \Pr[\hat{\mathcal{D}}(x) = 1]| \leq \sum_f 2\varepsilon_f \cdot d_f = 2\varepsilon$ .

Next, suppose that we seek a self-correction of  $\mathcal{D}$  that is  $\delta$ -close to a distribution over linear functions, for  $\delta \ll 2\varepsilon$ . One idea is to follow the same strategy as in the case of a single function: repeat the linearity test and then do self-correction. This idea, however, does not work now.

Consider the distribution  $\mathcal{D} = (1 - \varepsilon) \cdot \mathbf{0} + \varepsilon \cdot \mathbf{1}$ , i.e., the distribution that with probability  $1 - \varepsilon$  answers according to the all-zeros function (a linear function), and with probability  $\varepsilon$  according to the all-ones function (a function maximally far from linear functions). While  $\mathcal{D}$  passes the linearity test with probability  $1 - \varepsilon$ ,  $\mathcal{D}$  also passes the  $t$ -repeated linearity test with probability  $1 - \varepsilon$ . In other words, if  $\mathcal{D}$  passes the  $t$ -repeated linearity test with probability  $1 - \varepsilon$ , we can still only conclude that  $\hat{\mathcal{D}}$  is  $2\varepsilon$ -close to linear, independent of  $t$ .

While repeating the test does not increase the rejection probability, it can still be used to improve the quality of self-correction, by considering a *different* notion of self-correction that penalizes functions in the support of  $\mathcal{D}$  that are far from linear. Concretely, consider the distribution  $\mathcal{D}_t$  that equals  $\mathcal{D}$  when conditioned on the event that the  $t$ -repeated linearity test passes, and then define  $\hat{\mathcal{D}}_t$  to be the self-correction of  $\mathcal{D}_t$ . That is,  $\hat{\mathcal{D}}_t$  samples  $f$  from  $\mathcal{D}_t$  and answers any query  $x \in \{0,1\}^n$  by sampling  $z \in \{0,1\}^n$  and returning  $f(z+x) - f(z)$ . We claim that  $\hat{\mathcal{D}}_t$  is very close to linear.

Indeed, suppose that  $\mathcal{D}$  passes the  $t$ -repeated test with probability  $\gamma > 0$ , and let  $c > 1$  be a parameter. A function  $f$  sampled from  $\mathcal{D}_t$  is  $\frac{\ln c}{t}$ -close to linear with probability at least  $\frac{\gamma - 1/c}{\gamma} = 1 - \frac{1}{\gamma c}$ .<sup>9</sup> Setting  $c := t/\log t$ , the probability that  $\mathcal{D}_t$  outputs a function  $f$  that is  $\frac{\log t - \log \log t}{t}$ -far from linear is at most  $\frac{\log t}{\gamma t}$ . Therefore, by applying the argument from the beginning of this subsection, we conclude that  $\hat{\mathcal{D}}_t$  is  $O_\gamma(\frac{\log t}{t})$ -close to a distribution over linear functions.

We can further reduce the distance to be exponentially small in  $t$  by performing self-correction  $t$  times:  $\hat{\mathcal{D}}_t(x)$  now samples  $z_1, \dots, z_t \in \{0,1\}^n$ , and outputs the majority of  $\{\mathcal{D}(z_i + x) - \mathcal{D}(z_i)\}_{i \in [t]}$  conditioned on the event that the  $t$ -repeated linearity test passes. By setting  $c := 2^{t/10}$  in the discussion above, we conclude that if we sample  $f$  from  $\mathcal{D}_t$ , then  $f$  is 0.1-close to a linear function  $f^*$  with probability  $1 - \frac{1}{\gamma 2^{t/10}}$ . In particular, for every  $x \in \{0,1\}^n$  it holds that  $\Pr_{z_i}[f^*(z_i) = f(z_i) \wedge f^*(z_i + x) = f(z_i + x)] \geq 0.8$ , and so the probability that the majority value of  $\{\mathcal{D}(z_i + x) - \mathcal{D}(z_i)\}_{i \in [t]}$  is not equal to  $f^*(x)$  is  $2^{-O(t)}$ . In sum, the  $t$ -repeated self-correction conditioned on the event that the  $t$ -repeated linearity test passes yields us a distribution that is  $\frac{1}{\gamma} 2^{-O(t)}$ -close to linear.

<sup>8</sup> Recall that if  $f$  is 0.25-close to linear functions then  $f^*$  is unique. We do not rely on uniqueness.

<sup>9</sup> The  $t$ -repeated linearity test accepts a function  $f$  that is  $\frac{\ln c}{t}$ -far from linear with probability at most  $(1 - \frac{\ln c}{t})^t \leq \frac{1}{c}$ .

**The non-signaling case.** The case of non-signaling strategies is similar to the case of distributions in that the analysis of the self-correction involves conditioning over a certain event. Yet, the conclusions and steps of the proof are quite different. Informally, this is because non-signaling functions are quasi-distributions (probabilities can be negative), which prevents us from doing a straightforward analysis such as the one above. We now discuss how we address this.

Suppose that we have a  $k$ -non-signaling function  $\mathcal{F}: \{0, 1\}^n \rightarrow \{0, 1\}$  that passes the linearity test with probability  $1 - \varepsilon$ . The result of [9] proves that the self-correction  $\hat{\mathcal{F}}$  defined as  $\hat{\mathcal{F}}(x) := \mathcal{F}(z+x) - \mathcal{F}(z)$  (where  $z$  is chosen randomly from  $\{0, 1\}^n$ ) is  $2^{O(k)}\varepsilon$ -close to linear. This is too large in our setting as we have  $k = O(\log N)$ , and we would like the distance to be  $O(\varepsilon)$ . Instead, we prove a slightly different guarantee from [9]. Namely, we show that there is a linear non-signaling function  $\mathcal{L}$ , such that on every set  $S$ ,  $\hat{\mathcal{F}}$  is  $O(|S| \sqrt{\varepsilon})$ -close to  $\mathcal{L}$ . Unlike in the result of [9], our distance now decays with  $|S|$ , and is in particular independent of  $k$ . This is sufficient for our purposes, since we set  $|S| = 4$ , the number of queries made by the ALMSS verifier.

In our proof, we consider a *different* self-correction  $\bar{\mathcal{F}}_t$  that, unlike  $\hat{\mathcal{F}}$ , is *only used in the analysis* and is not used by the compiler. First, we show that  $\bar{\mathcal{F}}_t$  passes the linearity test with probability  $1 - \exp(-t)$ , and so the result of [9] implies that  $\bar{\mathcal{F}}_t$  is very close to a linear non-signaling function. Then, we relate  $\bar{\mathcal{F}}_t$  and  $\hat{\mathcal{F}}$  to show that  $\hat{\mathcal{F}}$  is  $O(\sqrt{\varepsilon})$ -close to a linear non-signaling function.

Informally, the self-correction  $\bar{\mathcal{F}}_t$  equals  $\mathcal{F}$  with the standard self-correction procedure repeated  $t$  times, conditioned on  $\mathcal{F}$  passing  $(1 - \sqrt{\varepsilon})t$  of  $t$  repetitions of the linearity test. In more detail, given a subset  $S \subseteq \{0, 1\}^n$ ,  $\bar{\mathcal{F}}_t(S)$  is the following distribution. For each  $x \in S$ , sample uniform and independent  $z_x^{(1)}, \dots, z_x^{(t)} \in \{0, 1\}^n$  conditioned on satisfying the same linear dependencies as in  $S$ ; for instance, if  $S = \{x, y, x+y\}$ , then  $z_x^{(i)} + z_y^{(i)} = z_{x+y}^{(i)}$  holds for all  $i$ . Then  $\bar{\mathcal{F}}_t$  assigns to each  $x \in S$  the value  $\text{MAJ}_{i \in [t]} \{\mathcal{F}(z_x^{(i)} + x) - \mathcal{F}(z_x^{(i)})\}$  *conditioned on the event* that  $\mathcal{F}$  passes at least  $(1 - \sqrt{\varepsilon})t$  of  $t$  repetitions of the basic linearity test. We note that if  $\mathcal{F}$  is linear, then  $\bar{\mathcal{F}}_t \equiv \hat{\mathcal{F}} \equiv \mathcal{F}$ .

The first part of the analysis uses a lemma that informally states that by conditioning on  $\mathcal{F}$  passing most of the  $t$ -repeated linearity tests, we force the conditioned  $\mathcal{F}$  to behave “close” to linear. Specifically, letting  $b_x^{(i)} = \mathcal{F}(z_i + x) - \mathcal{F}(z_i)$ , we get that with probability  $1 - \exp(-t)$  there is a bit  $b_x$  that equals  $b_x^{(i)}$  for at least  $\frac{3t}{4}$  of the  $i$ 's (so the majority is a vast majority), which implies that  $\bar{\mathcal{F}}_t(x) = b_x$ , and analogously for  $y$  and  $x+y$ . Then, via a similar argument, we show that with probability  $1 - \exp(-t)$  for at least  $\frac{3t}{4}$  of the  $i$ 's it holds that  $b_x^{(i)} + b_y^{(i)} = b_{x+y}^{(i)}$ . By union bound, these events hold simultaneously, and so we conclude that  $\bar{\mathcal{F}}_t$  satisfies  $\bar{\mathcal{F}}_t(x) + \bar{\mathcal{F}}_t(y) = \bar{\mathcal{F}}_t(x+y)$  with probability  $1 - \exp(-t)$ . We then invoke the result of [9] and conclude that  $\hat{\mathcal{F}}$  is very close to some linear non-signaling function  $\mathcal{L}$ .

In the second step, we relate  $\hat{\mathcal{F}}$  to  $\bar{\mathcal{F}}_t$  by claiming that if  $\Pr[\mathcal{F}(x) + \mathcal{F}(y) = \mathcal{F}(x+y)] \geq 1 - \varepsilon$ , then  $\hat{\mathcal{F}}$  and  $\bar{\mathcal{F}}_t$  are close in some precise sense. We first observe that if we run the  $t$ -repeated linearity test, i.e., choose  $x^{(1)}, y^{(1)}, \dots, x^{(t)}, y^{(t)}$  and check that  $\mathcal{F}(x^{(i)}) + \mathcal{F}(y^{(i)}) = \mathcal{F}(x^{(i)} + y^{(i)})$  for every  $i$ , then a simple Markov argument shows that with high probability, most of the linearity tests are satisfied. For instance, with probability  $1 - \sqrt{\varepsilon}$  at least  $(1 - \sqrt{\varepsilon})t$  of the  $i$ 's satisfy the linear constraint. This means that the event conditioned on in the definition of  $\bar{\mathcal{F}}_t$  is a large event. We also know from the first part of the analysis that, with high probability, the conditioning causes most of the evaluations of  $\mathcal{F}(z_x^{(i)} + x) - \mathcal{F}(z_x^{(i)})$  to output the same value. Intuitively, this implies that  $\hat{\mathcal{F}}$  is close to  $\bar{\mathcal{F}}_t$ , via the following reasoning. Since  $\bar{\mathcal{F}}_t$  conditions on a large event, it is close to the corresponding self-correction that does not condition at all. Since the majority taken over the evaluations of  $\mathcal{F}(z_x^{(i)} + x) - \mathcal{F}(z_x^{(i)})$

when computing  $\overline{\mathcal{F}}_t$  is a vast majority, with high probability  $\hat{\mathcal{F}}$  (which is a sample from one of the elements the majority is over) will agree with the vast majority. This allows us to conclude that for any set  $S$ ,  $\hat{\mathcal{F}}$  will be  $O(|S|\sqrt{\epsilon})$ -close to  $\overline{\mathcal{F}}_t$ .

See the full version of this paper for details.

---

## References

- 1 Sanjeev Arora, Carsten Lund, Rajeev Motwani, Madhu Sudan, and Mario Szegedy. Proof verification and the hardness of approximation problems. *Journal of the ACM*, 45(3):501–555, 1998. Preliminary version in FOCS '92.
- 2 Sanjeev Arora and Shmuel Safra. Probabilistic checking of proofs: a new characterization of NP. *Journal of the ACM*, 45(1):70–122, 1998. Preliminary version in FOCS '92.
- 3 László Babai, Lance Fortnow, Leonid A. Levin, and Mario Szegedy. Checking computations in polylogarithmic time. In *Proceedings of the 23rd ACM Symposium on Theory of Computing*, STOC '91, pages 21–32, 1991.
- 4 László Babai, Lance Fortnow, and Carsten Lund. Non-Deterministic Exponential Time has Two-Prover Interactive Protocols. *Computational Complexity*, 1:3–40, 1991. Preliminary version appeared in FOCS '90.
- 5 Mihir Bellare, Don Coppersmith, Johan Håstad, Marcos A. Kiwi, and Madhu Sudan. Linearity testing in characteristic two. *IEEE Transactions on Information Theory*, 42(6):1781–1795, 1996.
- 6 Eli Ben-Sasson, Oded Goldreich, Prahladh Harsha, Madhu Sudan, and Salil P. Vadhan. Robust PCPs of Proximity, Shorter PCPs, and Applications to Coding. *SIAM Journal on Computing*, 36(4):889–974, 2006.
- 7 Nir Bitansky and Alessandro Chiesa. Succinct Arguments from Multi-Prover Interactive Proofs and their Efficiency Benefits. In *Proceedings of the 32nd Annual International Cryptology Conference*, CRYPTO '12, pages 255–272, 2012.
- 8 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.
- 9 Alessandro Chiesa, Peter Manohar, and Igor Shinkar. Testing Linearity against Non-Signaling Strategies. In *Proceedings of the 33rd Annual Conference on Computational Complexity*, CCC '18, pages 17:1–17:37, 2018.
- 10 Irit Dinur and Omer Reingold. Assignment Testers: Towards a Combinatorial Proof of the PCP Theorem. In *Proceedings of the 45th IEEE Symposium on Foundations of Computer Science*, FOCS '04, pages 155–164, 2004.
- 11 Uriel Feige, Shafi Goldwasser, Laszlo Lovász, Shmuel Safra, and Mario Szegedy. Interactive proofs and the hardness of approximating cliques. *Journal of the ACM*, 43(2):268–292, 1996. Preliminary version in FOCS '91.
- 12 Justin Holmgren and Ron Rothblum. Delegation With (Nearly) Optimal Time/Space Overhead. In *Proceedings of the 59th IEEE Symposium on Foundations of Computer Science*, FOCS '18, pages ???–???, 2018.
- 13 Tsuyoshi Ito. Polynomial-Space Approximation of No-Signaling Provers. In *Proceedings of the 37th International Colloquium on Automata, Languages and Programming*, ICALP '10, pages 140–151, 2010.
- 14 Tsuyoshi Ito, Hirotada Kobayashi, and Keiji Matsumoto. Oracularization and Two-Prover One-Round Interactive Proofs against Nonlocal Strategies. In *Proceedings of the 24th IEEE Annual Conference on Computational Complexity*, CCC '09, pages 217–228, 2009.
- 15 Yael Kalai, Ran Raz, and Ron Rothblum. Delegation for Bounded Space. In *Proceedings of the 45th ACM Symposium on the Theory of Computing*, STOC '13, pages 565–574, 2013.



- 16 Yael Tauman Kalai, Ran Raz, and Oded Regev. On the Space Complexity of Linear Programming with Preprocessing. In *Proceedings of the 7th Innovations in Theoretical Computer Science Conference*, ITCS '16, pages 293–300, 2016.
- 17 Yael Tauman Kalai, Ran Raz, and Ron D. Rothblum. How to delegate computations: the power of no-signaling proofs. In *Proceedings of the 46th ACM Symposium on Theory of Computing*, STOC '14, pages 485–494, 2014. Full version available at <https://eccc.weizmann.ac.il/report/2013/183/>.
- 18 Leonid A Khalfin and Boris S Tsirelson. Quantum/classical correspondence in the light of Bell's inequalities. *Foundations of physics*, 22(7):879–948, 1992.
- 19 Joe Kilian. A note on efficient zero-knowledge proofs and arguments. In *Proceedings of the 24th Annual ACM Symposium on Theory of Computing*, STOC '92, pages 723–732, 1992.
- 20 Susumu Kiyoshima. No-signaling Linear PCPs. In *Proceedings of the 16th Theory of Cryptography Conference*, TCC '18, pages 67–97, 2018.
- 21 Silvio Micali. Computationally Sound Proofs. *SIAM Journal on Computing*, 30(4):1253–1298, 2000. Preliminary version appeared in FOCS '94.
- 22 Omer Paneth and Guy Rothblum. On Zero-Testable Homomorphic Encryption and Publicly Verifiable Non-Interactive Arguments. In *Proceedings of the 15th Theory of Cryptography Conference*, TCC '17, 2017.
- 23 Sandu Popescu and Daniel Rohrlich. Quantum nonlocality as an axiom. *Foundations of Physics*, 24(3):379–385, 1994.
- 24 Peter Rastall. Locality, Bell's theorem, and quantum mechanics. *Foundations of Physics*, 15(9):963–972, 1985.
- 25 Hanif D. Sherali and Warren P. Adams. A Hierarchy of Relaxations between the Continuous and Convex Hull Representations for Zero-One Programming Problems. *SIAM Journal on Discrete Mathematics*, 3(3):411–430, 1990.





# On the Algorithmic Power of Spiking Neural Networks

Chi-Ning Chou<sup>1</sup>

School of Engineering and Applied Sciences, Harvard University, Cambridge, MA 02138, USA  
chiningchou@g.harvard.edu

Kai-Min Chung

Institute of Information Science, Academia Sinica, Taipei, Taiwan  
kmchung@iis.sinica.edu.tw

Chi-Jen Lu

Institute of Information Science, Academia Sinica, Taipei, Taiwan  
cjl@iis.sinica.edu.tw

---

## Abstract

*Spiking Neural Networks* (SNN) are mathematical models in neuroscience to describe the dynamics among a set of neurons that interact with each other by firing instantaneous signals, *a.k.a.*, *spikes*. Interestingly, a recent advance in neuroscience [Barrett-Denève-Machens, NIPS 2013] showed that the neurons' *firing rate*, i.e., the average number of spikes fired per unit of time, can be characterized by the optimal solution of a quadratic program defined by the parameters of the dynamics. This indicated that SNN potentially has the computational power to solve non-trivial quadratic programs. However, the results were justified empirically without rigorous analysis.

We put this into the context of *natural algorithms* and aim to investigate the algorithmic power of SNN. Especially, we emphasize on giving rigorous asymptotic analysis on the performance of SNN in solving optimization problems. To enforce a theoretical study, we first identify a simplified SNN model that is tractable for analysis. Next, we confirm the empirical observation in the work of Barrett et al. by giving an upper bound on the convergence rate of SNN in solving the quadratic program. Further, we observe that in the case where there are infinitely many optimal solutions, SNN tends to converge to the one with smaller  $\ell_1$  norm. We give an affirmative answer to our finding by showing that SNN can solve the  $\ell_1$  minimization problem under some regular conditions.

Our main technical insight is a *dual view* of the SNN dynamics, under which SNN can be viewed as a new natural primal-dual algorithm for the  $\ell_1$  minimization problem. We believe that the dual view is of independent interest and may potentially find interesting interpretation in neuroscience.

**2012 ACM Subject Classification** Theory of computation → Models of computation

**Keywords and phrases** Spiking Neural Networks, Natural Algorithms,  $\ell_1$  Minimization

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2019.26

**Related Version** A full version of the paper is available at <https://arxiv.org/abs/1803.10375>.

**Acknowledgements** The authors would like to thank Tsung-Han Lin, Zhenming Liu, Luca Trevisan, Richard Peng, Yin-Hsun Huang, and Tao Xiao for useful discussions related to this paper. We are also thankful to the anonymous reviewer from ITCS 2019 for useful comments.

---

<sup>1</sup> Supported by NSF awards CCF 1565264 and CNS 1618026.



## 1 Introduction

The theory of *natural algorithms* is a framework that bridges the algorithmic thinking in computer science and the mathematical models in biology. Under this framework, biological systems are viewed as *algorithms* to *efficiently* solve specific *computational problems*. Seminal works such as bird flocking [16, 17], slime systems [51, 65, 10], and evolution [38, 37] successfully provide algorithmic explanations for different natural objects. These works give rigorous theoretical results to confirm empirical observations, shed new light on the biological systems through computational lens, and sometimes lead to new biologically inspired algorithms.

In this work, we investigate *Spiking Neural Networks (SNNs)* as natural algorithms for solving convex optimization problems. SNNs are mathematical models for biological neural networks where a network of neurons transmit information by *firing spikes* through their synaptic connections (i.e., edges between two neurons). Our starting point is a seminal work of Barrett, Denève, and Machens [4], where they showed that the *firing rate* (i.e., the average number of spikes fired by each neuron) of a certain class of *integrate-and-fire* SNNs can be characterized by the optimal solutions of a quadratic program defined by the parameters of SNN. Thus, the SNN can be viewed as a natural algorithm for the corresponding quadratic program. However, no rigorous analysis was given in their work.

We bridge the gap by showing that the firing rate converges to an optimal solution of the corresponding quadratic program with an explicit polynomial bound on the convergent rate. Thus, the SNN indeed gives an *efficient algorithm* for solving the quadratic program. To the best of our knowledge, this is the first result with an explicit bound on the convergent rate. Previous works [58, 59, 63] on related SNN models for optimization problems are either heuristic or only proving convergence results when the time goes to infinity (see Section 1.4 for full discussion on related works).

We take one step further to ask what other optimization problems can SNNs efficiently solve. As our main result, we show that when configured properly, SNNs can solve the  $\ell_1$  *minimization problem*<sup>2</sup> in polynomial time<sup>3</sup>. Our main technical insight is interpreting the dynamics of SNNs in a *dual space*. In this way, SNNs can be viewed as a new primal-dual algorithm for solving the  $\ell_1$  *minimization problem*.

In the rest of the introduction, we will first briefly introduce the background of spiking neural networks (SNNs) and formally define the mathematical model we are working on. Next, our results will be presented and compared with other related works. Finally, we wrap up this section with potential future research directions and perspectives.

### 1.1 Spiking Neural Networks

Spiking neural networks (SNNs) are mathematical models for describing the dynamics of biological neural networks. An SNN consists of neurons, and each of them is associated with an intrinsic electrical charge called *membrane potential*. When the potential of a neuron reaches a certain level, it will fire an instantaneous signal, i.e., *spike*, to other neurons and increase or decrease their potentials.

---

<sup>2</sup> The problem is defined as given  $A \in \mathbb{R}^{m \times n}$ ,  $\mathbf{b} \in \mathbb{R}^m$ , and guaranteed that there is a solution to  $A\mathbf{x} = \mathbf{b}$ . The goal is finding a solution  $\mathbf{x}$  with the smallest  $\ell_1$  norm. See Section 2 for formal definition.

<sup>3</sup> The running time is polynomial in a parameter depending on the inputs. In some cases, this parameter might cause the running to be quasi-polynomial or sub-exponential. See the full version of this paper on arXiv for more details.

Mathematically, the dynamic of neuron's membrane potential in an SNN is typically described by a *differential equation*, and there are many well-studied models such as the *integrate-and-fire model* [35], the *Hodgkin-Huxley model* [27], and their variants [21, 61, 49, 26, 23, 33, 14, 22, 29, 64]. In this work, we focus on the integrate-and-fire model defined as follows. Let  $n$  be the number of neurons and  $\mathbf{u}(t) \in \mathbb{R}^n$  be the vector of membrane potentials where  $\mathbf{u}_i(t)$  is the potential of neuron  $i$  at time  $t$  for any  $i \in [n]$  and  $t \geq 0$ . The dynamics of  $\mathbf{u}(t)$  can be described by the following differential equation: for each  $i \in [n]$  and  $t \geq 0$

$$\frac{d}{dt} \mathbf{u}_i(t) = \sum_{j \in [n]} -C_{ji}(t) \mathbf{s}_j(t) + \mathbf{I}_i(t) \quad (1)$$

where the initial value of the potentials are set to 0, i.e.,  $\mathbf{u}_i(0) = 0$  for each  $i \in [n]$ . There are two terms that determine the dynamics of membrane potentials as shown in (1). The simpler term is the input charging<sup>4</sup>  $\mathbf{I}(t) \in \mathbb{R}^n$ , which can be thought of as an external effect on each neuron. The other term models the instantaneous spike effect among neurons. Specifically, the  $-C_{ji}(t) \mathbf{s}_j(t)$  term models the effect on the potential of neuron  $i$  when neuron  $j$  fires a spike. Here  $C(t) \in \mathbb{R}^{n \times n}$  is the connectivity matrix that encodes the *synapses* between neurons, where  $C_{ji}(t)$  describes the connection strength from neuron  $j$  to neuron  $i$ .  $\mathbf{s}(t) \in \mathbb{R}^n$  is the *spike train* which records the spikes of each neuron, and  $\mathbf{s}_i(t)$  can be thought of as indicating whether neuron  $i$  fires a spike at time  $t$ . To sum up, the  $-C_{ji}(t) \mathbf{s}_j(t)$  term decreases<sup>5</sup> the potential of neuron  $i$  by  $C_{ji}(t^*)$  whenever neuron  $j$  fires a spike at time  $t^*$ .

The spike train  $\mathbf{s}(t)$  is determined by the spike events, which are in turn determined by the spiking rule. A typical spiking rule is the threshold rule. Specifically, let  $\eta > 0$  be the spiking threshold, the threshold rule simply says that neuron  $i$  fires a spike at time  $t$  if and only if  $\mathbf{u}_i(t) > \eta$ . Next, record the timings when neuron  $i$  fires a spike as  $0 \leq t_1^{(i)} < t_2^{(i)} < \dots$  and let  $k_i(t)$  be the number of spikes within time  $[0, t]$ . An important statistics of the dynamics is the *firing rate* defined as  $\mathbf{x}_i(t) := k_i(t)/t$  for neuron  $i \in [n]$  at time  $t$ , namely, the average number of spikes of neuron  $i$  up to time  $t$ . The last thing we need for specifying  $\mathbf{s}(t)$  is the *spike shape*, which can be modeled as a function  $\delta : \mathbb{R}_{\geq 0} \rightarrow \mathbb{R}$ . Intuitively, the spike shape describes the effect of a spike, and standard choices of  $\delta$  could be the Dirac delta function or a pulse function with an exponential tail. Now we can define  $\mathbf{s}_i(t) = \sum_{1 \leq s \leq k_i(t)} \delta(t - t_s^{(i)})$  to be the *spike train* of neuron  $i$  at time  $t$ .

We provide the following example to illustrate the SNN dynamics introduced above.

► **Example 1.** Let  $n = 2$ ,  $\eta = 1$ , and  $\delta$  be the Dirac delta function such that for any  $\epsilon > 0$ ,  $\int_0^\epsilon \delta(t) dt = 1$  and  $\delta(t) \geq 0$  for any  $t \geq 0$ . Let both external charging and connectivity matrix be static, i.e.,  $\mathbf{I}(t) = \mathbf{I}$  and  $C(t) = C$  for any  $t \geq 0$ , and consider

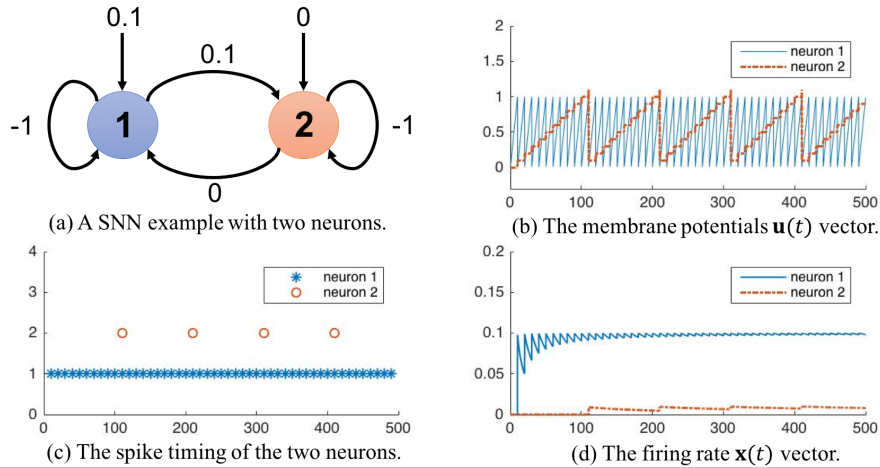
$$C = \begin{pmatrix} 1 & 0 \\ -0.1 & 1 \end{pmatrix}, \quad \mathbf{I} = \begin{pmatrix} 0.1 \\ 0 \end{pmatrix}, \quad \text{and } \mathbf{u}(0) = \begin{pmatrix} 0 \\ 0 \end{pmatrix}.$$

In Figure 1, we simulate this SNN for 500 seconds. We can see that neuron 1 fires a spike every ten seconds while neuron 2 fires a spike every one hundred seconds. As a result, the firing rate of neuron 1 will gradually converge to 0.1 and that of neuron 2 will go to 0.01.

In general, both the input charging vector  $\mathbf{I}(t)$  and the connectivity matrix  $C(t)$  can evolve over time, in which the change of  $\mathbf{I}(t)$  models the variation of the environment and the change of  $C_{ji}(t)$  captures the adaptive *learning* behavior of the neurons to the environmental change.

<sup>4</sup> Also known as *input signal* or *input current*.

<sup>5</sup> If  $C_{ji}(t^*) < 0$ , then the potential of neuron  $i$  actually *increases* by  $|C_{ji}(t^*)|$ .



■ **Figure 1** The example of SNN with two neurons. In (a), we describe the dynamic of this SNN. Note that the effect of spikes is the negation of the synapse encoded in the connectivity matrix  $C$ . In (b), we plot the membrane potential vectors  $\mathbf{u}(t)$ . In (c), we plot the timings when neurons fire a spike. One can see that neuron 1 fires a spike every ten seconds while neuron 2 fires a spike every one hundred seconds. In (d), we plot the firing rate vector  $\mathbf{x}(t)$ . One can see that the firing rate of neuron 1 will gradually converge to 0.1 and that of neuron 2 will go to 0.01.

Understanding how synapses evolve over time (i.e., synapse plasticity) is a very important subject in neuroscience. However, in this work, we follow the choice of Barrett et al. [4] and consider *static* SNN dynamics, where both the input charging  $\mathbf{I}(t)$  and the synapses  $C(t)$  are constants. Although this is a special case compared to the general model in (1), we justify the choice of static SNN by showing that SNN already exhibits non-trivial computational power even in this restricted model.

As in Barrett et al. [4], we focus on static SNN and view it as a natural algorithm for optimization problems. Specifically, given an instance to the optimization problem, the goal is to configure a static SNN (by setting its parameters) so that the firing rate converge to an optimal solution efficiently. In this sense, the result of Barrett et al. [4] can be interpreted as a natural algorithm for certain quadratic programs. In our eyes, the solution being encoded as the firing rate is an interesting and peculiar feature of the SNN dynamics. Also, the dynamics of a static SNN can be viewed as a simple distributed algorithm with a simple communication pattern. Specifically, once the dynamics is set up, each neuron only needs to keep track of its potential and communicate with each other through spikes.

## 1.2 Our Results

Barrett et al. [4] gave a clean characterization of the firing rates by the network connectivity and input signal. Concretely, they consider *static* SNN where both the connectivity matrix  $C \in \mathbb{R}^{n \times n}$  and the external charging  $\mathbf{I} \in \mathbb{R}^n$  do not change with time. They argued that the firing rate would converge to the solution of the following quadratic program.

$$\begin{aligned}
 & \underset{\mathbf{x} \in \mathbb{R}^n}{\text{minimize}} && \|C\mathbf{x} - \mathbf{I}\|_2^2 \\
 & \text{subject to} && \mathbf{x}_i \geq 0, \forall i \in [n].
 \end{aligned} \tag{2}$$

They supported this observation by giving simulations on the so called *tightly balanced networks* and yielded pretty accurate predictions in practice. Also, they heuristically explained the reason how they came up with the quadratic program. However, no rigorous theorem had been proved on the convergence of firing rate to the solution of this quadratic program.

To give a theoretical explanation for the discovery of [4], we start with a simpler SNN model to enable the analysis.

**The simple SNN model.** In the simple SNN model, we make two simplifications on the general model in (1).

First, we pick the shape of spike to be the Dirac delta function. That is, let  $\delta(t) = \mathbf{1}_{t=0}$  and thus  $\mathbf{s}_i(t) = \mathbf{1}_{\mathbf{u}_i(t) > \eta}$ . This simplification saves us from complicated calculation while the Dirac delta function still captures the instantaneous behavior of a spike.

Second, we consider the connectivity matrix  $C$  in the form  $C = \alpha \cdot A^\top A$  where  $\alpha > 0$  is the spiking strength and  $A \in \mathbb{R}^{m \times n}$  is the Cholesky decomposition of  $C$ . The reason for introducing  $\alpha$  is to model the height of the Dirac delta function. Mathematically, it is redundant to have both  $\alpha$  and  $C$  since the model remains the same when combining  $\alpha$  with  $C$ . However, as we will see in the next subsection, separating  $\alpha$  and  $C$  is meaningful as  $C$  corresponds to the *input* of the computational problem and  $\alpha$  is the parameter that one can choose to configure an SNN to solve the problem.

In this work, we focus on the algorithmic power of SNN in the following sense. Given a problem instance, one configures a SNN and sets the firing rate  $\mathbf{x}(t)$  to be the output at time  $t$ . We say this SNN solves the problem if  $\mathbf{x}(t)$  converges to the solution of the problem.

**Simple SNN solves the non-negative least squares.** As mentioned, Barrett et al. [4] identified a connection between the firing rate of SNN with integrate-and-fire neurons and a quadratic programming problem (2). They gave empirical evidence for the correctness of this connection, however, no theoretical guarantee had been provided. Our first result confirms their observation by giving the first theoretical analysis. Specifically, when  $C = A^\top A$  and  $\mathbf{I} = A^\top \mathbf{b}$ , the firing rate will converge to the solution of the following *non-negative least squares problem*.

$$\begin{aligned} & \underset{\mathbf{x} \in \mathbb{R}^n}{\text{minimize}} && \|\mathbf{A}\mathbf{x} - \mathbf{b}\|_2^2 \\ & \text{subject to} && \mathbf{x}_i \geq 0, \forall i \in [n]. \end{aligned} \tag{3}$$

► **Theorem 1 (informal).** *Given  $A \in \mathbb{R}^{m \times n}$ ,  $\mathbf{b} \in \mathbb{R}^m$ , and  $\epsilon > 0$ . Suppose  $A$  satisfies some regular conditions<sup>6</sup>. Let  $\mathbf{x}(t)$  be the firing rate of the simple SNN with  $0 < \alpha \leq \alpha(A)$  where  $\alpha(A)$  is a function depending on  $A$ . When  $t \geq \Omega(\frac{\sqrt{n}}{\epsilon \|\mathbf{b}\|_2})$ ,<sup>7</sup>  $\mathbf{x}(t)$  is an  $\epsilon$ -approximate solution<sup>8</sup> for the non-negative least squares problem of  $(A, \mathbf{b})$ .*

The formal statement and the proof for the theorem are provided in the Section 4 of the full version of this paper. To the best of our knowledge, this is the first<sup>9</sup> theoretical result on the analysis of SNN with an explicit bound on the convergence rate and shows that SNN can be implemented as an efficient algorithm for an optimization problem.

<sup>6</sup> More details about the regular conditions will be discussed in Section 3.3 of the full version.

<sup>7</sup> The  $\Omega(\cdot)$  and the  $O(\cdot)$  later both hide the dependency on some parameters of  $A$ . See Section 3.3 of the full version for more details.

<sup>8</sup> See Definition 3 for the formal definition of  $\epsilon$ -approximate solution.

<sup>9</sup> See Section 1.4 for comparisons with related works.

**Simple SNN solves the  $\ell_1$  minimization problem.** In addition to solving the non-negative least squares problem, as our main result, we also show that the simple SNN is able to solve the  $\ell_1$  *minimization problem*, which is defined as minimizing the  $\ell_1$  norm of the solutions of  $A\mathbf{x} = \mathbf{b}$ .  $\ell_1$  minimization problem is also known as the *basis pursuit* problem proposed by Chen et al. [18]. The problem is widely used for recovering sparse solution in compressed sensing, signal processing, face recognition etc.

Before the discussion on  $\ell_1$  minimization, let us start with a digression on the *two-sided* simple SNN for the convenience of future analysis.

$$\frac{d}{dt}\mathbf{u}(t) = -\alpha \cdot A^\top A\mathbf{s}(t) + A^\top \mathbf{b}$$

where  $\mathbf{s}_i(t) = \mathbf{1}_{\mathbf{u}_i(t) > \eta} - \mathbf{1}_{\mathbf{u}_i(t) < -\eta}$ . Note that the two-sided SNN is a special case of the one-sided SNN in the sense that one can use the one-sided SNN to simulate the two-sided SNN as follows. Given a two-sided SNN described above with connectivity matrix  $C = A^\top A$  and external charging  $\mathbf{I} = A^\top \mathbf{b}$ . Let  $C' = \begin{pmatrix} A^\top A & -A^\top A \\ -A^\top A & A^\top A \end{pmatrix}$  and  $\mathbf{I}' = \begin{pmatrix} A^\top \mathbf{b} \\ -A^\top \mathbf{b} \end{pmatrix}$ . Intuitively, this can be thought of as duplicating each neuron and flip its connectivities with other neurons.

To solve the  $\ell_1$  minimization problem, we simply configure a two-sided SNN as follows. Given an input  $(A, \mathbf{b})$ , let  $C = A^\top A$  and  $\mathbf{I} = A^\top \mathbf{b}$ . Now, we have the following theorem.

► **Theorem 2 (informal).** *Given  $A \in \mathbb{R}^{m \times n}$ ,  $\mathbf{b} \in \mathbb{R}^m$ , and  $\epsilon > 0$ . Suppose  $A$  satisfies some regular conditions. Let  $\mathbf{x}(t)$  be the firing rate of the two-sided simple SNN with  $0 < \alpha \leq \alpha(A)$  where  $\alpha(A)$  is a function depending on  $A$ . When  $t \geq \Omega(\frac{n^3}{\epsilon^2})$ ,  $\mathbf{x}(t)$  is an  $\epsilon$ -approximate solution<sup>10</sup> for the  $\ell_1$  minimization problem of  $(A, \mathbf{b})$ .*

See Theorem 7 for the formal statement of this theorem. As we will discuss in the next subsection, under the dual view of the SNN dynamics, the simple two sided SNN can be interpreted as a new natural primal-dual algorithm for the  $\ell_1$  minimization problem.

### 1.3 A Dual View of the SNN Dynamics

The main techniques in this work is the discovery of a *dual view* of SNN. Recall that the dynamic of a static SNN can be described by the following differential equation.

$$\frac{d}{dt}\mathbf{u}(t) = -\alpha \cdot C\mathbf{s}(t) + \mathbf{I}$$

where  $\mathbf{u}(0) = \mathbf{0}$  the parameters  $C$  and  $\mathbf{I}$  can be represented as  $C = A^\top A$  and  $\mathbf{I} = A^\top \mathbf{b}$  for some  $A \in \mathbb{R}^{m \times n}$  and  $\mathbf{b} \in \mathbb{R}^m$ . For simplicity, we pick the firing threshold  $\eta = 1$  here. Let us call the dynamics of  $\mathbf{u}(t)$  the *primal SNN*. Now, the *dual SNN*, can be defined as follows.

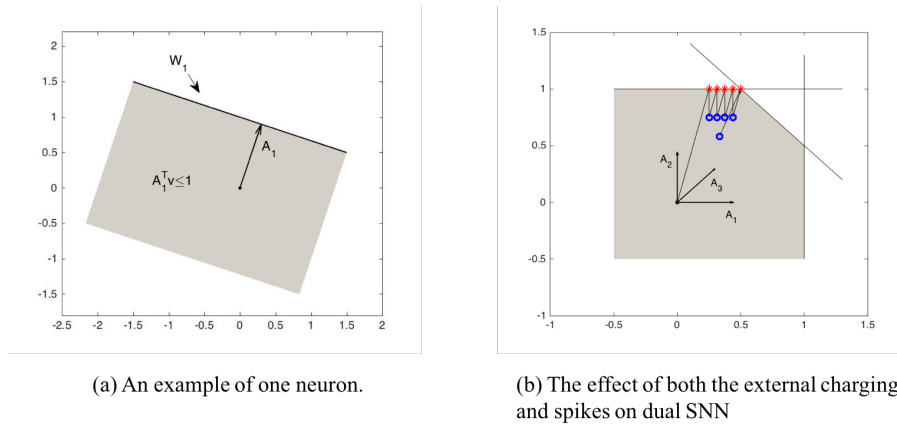
$$\frac{d}{dt}\mathbf{v}(t) = -\alpha \cdot A\mathbf{s}(t) + \mathbf{b}$$

where  $\mathbf{v}(0) = \mathbf{0}$  and  $\mathbf{s}(t)$  defined as the usual way. At first glance, this merely looks like a simple linear transformation, Nevertheless, the dual SNN provides a nice *geometric view* for the SNN dynamics as follows.

At each update in the dynamics, there are two terms affecting the dual SNN  $\mathbf{v}(t)$ : the external charging  $\mathbf{b} \cdot dt$  and the spiking effect  $-\alpha \cdot A\mathbf{s}(t)$ . First, the external charging  $\mathbf{b} \cdot dt$  can be thought of as a constant force that drags that dual SNN in the direction  $\mathbf{b}$ .

<sup>10</sup>See Definition 4 for the formal definition of  $\epsilon$ -approximate solution.





■ **Figure 2** These are examples of the geometric interpretation of the dual SNN. In (a), we have one neuron where  $A_1 = [\frac{1}{2} \ 1]^T$ . In this case, neuron  $i$  would not fire as long as the dual SNN  $\mathbf{v}(t)$  stays in the gray area. In (b), we consider a SNN with 3 neurons where  $A_1 = [1 \ 0]^T$ ,  $A_2 = [0 \ 1]^T$ , and  $A_3 = [\frac{2}{3} \ \frac{2}{3}]^T$ . One can see that the effect of spikes on dual SNN is a jump in the direction of the normal vector of the wall(s).

■ **Table 1** Comparison of the geometric view of primal and dual SNNs.

	Primal SNN $\mathbf{u}(t)$	Dual SNN $\mathbf{v}(t)$
Spiking rule	$\mathbf{u}_i(t) > 1$	$A_i^T \mathbf{v}(t) > 1$
Spiking effect	$-\alpha \cdot A^T A_i$	$-\alpha \cdot A_i$

To explain the effect of spikes in the dual view, let us start with an geometric view for the spiking rule. Recall that neuron  $i$  fires a spike at time  $t$  if and only if  $\mathbf{u}_i(t) > 1$ . In the language of dual SNN, this condition is equivalent to  $A_i^T \mathbf{v}(t) > 1$ . Let  $W_i = \{\mathbf{v} \in \mathbb{R}^m : A_i^T \mathbf{v} = 1\}$  be the *wall* of neuron  $i$ , the above observation is saying that neuron  $i$  will fire a spike once it penetrates the wall  $W_i$  from the half-space  $\{\mathbf{v} \in \mathbb{R}^m : A_i^T \mathbf{v} \leq 1\}$ . See Figure 2 for an example. After neuron  $i$  fires a spike, the spiking effect on the dual SNN  $\mathbf{v}(t)$  would be a  $-\alpha \cdot A_i$  term, which corresponds to a jump in the *normal direction* of  $W_i$ . See Figure 2 for an example.

The geometric interpretation described above is the main advantage of using dual SNN. Specifically, this gives us a clear picture of how spikes affect the SNN dynamics. Namely, neuron  $i$  fires a spike if and only if the dual SNN  $\mathbf{v}(t)$  penetrates the wall  $W_i$  and then  $\mathbf{v}(t)$  jumps back in the normal direction of  $W_i$ . Note that this connection would not hold in the primal SNN. In primal SNN  $\mathbf{u}(t)$ , neuron  $i$  fires a spike if and only if  $\mathbf{u}_i(t) > 1$  while the effect on  $\mathbf{u}(t)$  is moving in the direction  $-A^T A_i$ . See Table 1 for a comparison.

**Dual view of SNN as a primal-dual algorithm for  $\ell_1$  minimization problem.** First, let us write down the  $\ell_1$  minimization problem and its dual.

$$\begin{array}{ll}
 \underset{\mathbf{x} \in \mathbb{R}^n}{\text{minimize}} & \|\mathbf{x}\|_1 \\
 \text{subject to} & \mathbf{A}\mathbf{x} = \mathbf{b}.
 \end{array}
 \qquad
 \begin{array}{ll}
 \underset{\mathbf{v} \in \mathbb{R}^m}{\text{maximize}} & \mathbf{b}^T \mathbf{v} \\
 \text{subject to} & \|A^T \mathbf{v}\|_\infty \leq 1.
 \end{array}$$

Now we observe that the dual dynamics can be viewed as a variant of the projected gradient descent algorithm to solve the dual program. Before the explanation, recall that for the  $\ell_1$  minimization problem, we are considering the two-sided SNN for convenience. Indeed, without the spiking term,  $\mathbf{v}(t)$  simply moves towards the gradient direction  $\mathbf{b}$  of the dual objective function  $\mathbf{b}^\top \mathbf{v}$ . For the spike term  $-\alpha \cdot A\mathbf{s}(t)$ , note that  $\mathbf{s}_i(t) \neq 0$  (i.e., neuron  $i$  fires) if and only if  $|A_i^\top \mathbf{v}(t)| = |\mathbf{u}_i(t)| > 1$ , which means that  $\mathbf{v}(t)$  is outside the feasible polytope  $\{\mathbf{v} : \|A^\top \mathbf{v}\|_\infty \leq 1\}$  of the dual program. Therefore, one can view the role of the spike term as *projecting*  $\mathbf{v}(t)$  back to the feasible polytope. That is, when the dual SNN  $\mathbf{v}(t)$  becomes infeasible, it triggers some spikes, which maintains the dual feasibility and updates the primal solution (the firing rate). To sum up, we can interpret the simple SNN as performing a non-standard projected gradient descent algorithm for the dual program of  $\ell_1$  minimization in the dual view of SNN.

With this primal-dual view in mind, we analyze the SNN algorithm by combining tools from convex geometry and perturbation theory as well as several non-trivial structural lemmas on the geometry of the dual program of  $\ell_1$  minimization. One of the key ingredients here is identifying a *potential function* that (i) upper bounds the error of solving  $\ell_1$  minimization problem and (ii) monotonously converges to 0. More details will be provided in Section 3.

## 1.4 Related Work

We compare this research with other related works in the following four aspects.

**Computational power of SNN.** Recognized as the third generation of neural networks [45], the theoretical foundation for the computability of SNN had been built in the pioneering works of Maass et al. [43, 45, 46, 48] in which SNN was shown to be able to simulate standard computational models such as Turing machines, random access machines (RAM), and threshold circuits.

However, this line of works focused on the universality of the computational power and did not consider the efficiency of SNN in solving specific computational problems. In recent years, a line of exciting research have reported the efficiency of SNN in solving specific computational problems such as sparse coding [68, 62, 63], dictionary learning [36], pattern recognition [19, 32, 6], and quadratic programming [4]. These works indicated the advantage of SNN in handling *sparsity* as well as being *energy efficient* and inspired real-world applications [5]. However, to the best of our knowledge, no theoretical guarantee on the efficiency of SNN had been provided. For instance, Tang et al. [62, 63] only proved the *convergence in the limit* result for SNN solving sparse coding problem instead of giving an explicit convergence rate analysis. The main contribution in this work is giving a rigorous guarantee on the convergence rate of the computational power of SNN.

**The number of spikes versus the timing of spikes.** In this work, we mainly focus on the firing rate of SNN. That is, we only study the computational power with respect to the *number* of spikes. Another important property of SNN is the *timing* of spikes.

The power of the timing of spikes had been reported since the 90s from some experimental evidences indicating that neural systems might use the timing of spikes to encode information [1, 28, 54]. From then on, a bunch of works have been focused on the aspect of time as a basis of information coding both from theoretical [52, 45, 48, 66] and experimental [25, 7, 34] sides. It is generally believed that the timing of spikes is more powerful than the firing rate [67, 56, 53]. Other than the capacity of encoding information, the timing of spikes has also been studied in the context of computational power [67, 44, 45, 24] and learning [12, 3, 60]. See the survey by Paugam et al. [53] for a thorough discussion.

While the timing of spikes is conceived as an important source of the power of SNN, in this work we simply focus on the firing rate and already yield some non-trivial findings in terms of the computational power. We believe that our work is still in the very beginning stage of the study of the computational power of SNN. Investigating how does the timing of spikes play a role is an interesting and important future direction. Immediate open questions here would be how could the timing of spikes fit into our study? What's the dual view of the timing of spikes? Can the timing of spikes solve the optimization problems more efficiently? Can the timing of spikes solve more difficult problems?

**SNN with randomness.** While most of the literature focused on deterministic SNN, there is also an active line of works studying the SNN model with randomness<sup>11</sup> [2, 57, 20, 15, 30, 47, 31, 40, 41, 42, 39].

Buesing et al. [15] used *noisy SNN* to implement MCMC sampling and Jonke et al. [30, 47, 31] further instantiated the idea to attack **NP**-hard problems such as *traveling salesman problem (TSP)* and *constraint satisfaction problem (CSP)*. Concretely, their noisy SNN has a randomized spiking rule and the firing pattern would form a distribution over the solution space whereas the closer a solution is to the optimal solution, the higher the probability it is sampled. They got nice experimental performance in terms of solving empirical instance approximately. They also pointed out that their noisy SNN has the potential to be implemented energy-efficiently in practice.

Lynch, Musco, and Parter [41] studied the *stochastic SNNs* with a focus on the Winner-Take-All (WTA) problem. Their sequence of works [40, 41, 42, 39] gave the first asymptotic analysis for stochastic SNN in solving WTA, similarity testing, and neural coding. They view SNNs as distributed algorithms and derived computational tradeoff in running time and network size.

In this work, we consider the SNN model without randomness and thus is incomparable with the above SNN models with randomness. It is an interesting direction to apply the dual view of deterministic SNN to SNN with randomness.

**Locally competitive algorithms.** Inspired by the dynamics of biological neural networks, Ruzell et al. designed the *locally competitive algorithms (LCA)* [55] for solving the Lasso (least absolute shrinkage and selection operator) optimization problem<sup>12</sup>, which is widely used in statistical modeling. Roughly speaking, LCA is also a dynamics among a set of *artificial neurons* which continuously signal their potential values (or a function of the values) to their neighboring neurons. There are two main differences between SNN and LCA. First, the neuron in SNN fires discrete spikes while the artificial neuron in LCA produces continuous signal. Next, the neurons' potentials in LCA will converge to a fixed value, which is the output of the algorithm. In contrast, in SNN, only the neurons' firing rates may converge instead of their potentials.

Nevertheless, there is a *spikified* version of LCA introduced by Shapero et al. [58, 59] called *spike LCA (S-LCA)* in which the continuous signals are replaced with discrete spikes. S-LCA is almost the same as the SNN we are considering except a shrinkage term<sup>13</sup>. Recently, Tang et al. [63] showed that the firing rate of S-LCA indeed converges to a variant of

<sup>11</sup> SNN model with noise is also known as stochastic SNN or noisy SNN depending on how the randomness involves in the model.

<sup>12</sup> Note that Lasso is equivalent to the Basis Pursuit De-Noising (BPDN) program under certain parameters transformation.

<sup>13</sup> That is, the potential of each neuron will drop with rate proportional to the current potential value.

Lasso problem<sup>14</sup> in the limit. These works also experimentally demonstrated the efficient convergence of S-LCA and its advantage of fast identifying sparse solutions with potentially competitive practical performance to other Lasso algorithms (e.g., FISTA [5]). However, there is no proof of convergence rate, and thus no explicit complexity bound of S-LCA.

## 1.5 Future Works and Perspectives

In this work, we give a theoretical study on the algorithmic power of SNN. Specifically, we focus on the firing rate of SNN and confirm an empirical analysis by Barrett et al. [4] with a convergence theorem (i.e., Theorem 1). Furthermore, we discover a dual view of SNN and show that SNN is able to solve the  $\ell_1$  minimization problem (i.e., Theorem 2). In the following, we give interpretations to our results and point out future research directions.

First, how to interpret the dual dynamics of SNN? In this work, we discover the dual SNN based on mathematical convenience. Is there any biological interpretation?

Second, push further the analysis of simple SNN. We believe the parameters we get in the main theorems are not optimal. Is it possible to further sharpen the upper bound? We think this is both theoretically and practically interesting because both non-negative least squares and  $\ell_1$  minimization are important problems that have been well-studied in the literature. Comparing the running time complexity or parallel time complexity of SNN algorithm with other algorithms could also be of theoretical interest and might inspire new algorithm with better complexity. Also, for practical purpose, having better parameters would give more confidence in implementing SNN as a natural algorithm.

Third, further investigate the potential of SNN dynamics as natural algorithms. The question is two-folded: (i) What algorithms can SNN implement? (ii) What computational problems can SNN solve? It seems that SNN is good at dealing with sparsity. Could it be helpful in related computational tasks such as fast Fourier transform (FFT) or sparse matrix-vector multiplication? It is interesting to identify optimization problems and class of instances where SNN algorithm can outperform other algorithms.

Finally, explore the practical advantage of SNN dynamics as natural algorithms. The potential practical time efficiency, energy efficiency, and simplicity for hardware implementation have been suggested in several works [50, 8, 9]. It would be exciting to see whether SNN has nice performance on practical applications such as compressed sensing, Lasso, and etc.

**Organization.** The rest of the paper is organized as follows. Preliminaries are provided in Section 2. In Section 3, we formally present the dual view of SNN and give a proof sketch for the convergence theorem for  $\ell_1$  minimization problem. The full proofs for Theorem 1 and Theorem 2 are provided in the full version of this paper available on arXiv.

## 2 Preliminaries

In Section 2.1, we build up some notations for the rest of the paper. In Section 2.2, we define two optimization problems and the corresponding convergence guarantees.

### 2.1 Notations

For any  $n \in \mathbb{N}$ , denote  $[n] = \{1, 2, \dots, n\}$  and  $[\pm n] = \{\pm 1, \pm 2, \dots, \pm n\}$ . Let  $\mathbf{x}, \mathbf{y} \in \mathbb{R}^n$  be two vectors.  $|\mathbf{x}| \in \mathbb{R}^n$  denotes the entry-wise absolute value of  $\mathbf{x}$ , i.e.,  $|\mathbf{x}|_i = |\mathbf{x}_i|$  for any  $i \in [n]$ .  $\mathbf{x} \preceq \mathbf{y}$  refers to entry-wise comparison, i.e.,  $\mathbf{x}_i \leq \mathbf{y}_i \forall i \in [n]$ .

<sup>14</sup>In this variant, all the entries in matrix  $A$  is non-negative.

Let  $A$  be an  $m \times n$  real matrix. For any  $i \in [n]$ , denote the  $i$ th column of  $A$  as  $A_i$  and its negation to be  $A_{-i}$ , i.e.,  $A_{-i} = -A_i$ . When  $A$  is positive semidefinite, we define the  $A$ -norm of a vector  $\mathbf{x} \in \mathbb{R}^n$  to be  $\|\mathbf{x}\|_A := \sqrt{\mathbf{x}^\top A \mathbf{x}}$ . Let  $A^\dagger$  to be the pseudo-inverse of  $A$ . Define the maximum eigenvalue of  $A$  as  $\lambda_{\max}(A) := \max_{\mathbf{x} \in \mathbb{R}^n: \|\mathbf{x}\|_2=1} \|\mathbf{x}\|_A$ , the minimum non-zero eigenvalue of  $A$  to be  $\lambda_{\min}(A) := 1/(\max_{\mathbf{x} \in \mathbb{R}^n: \|\mathbf{x}\|_2=1} \|\mathbf{x}\|_{A^\dagger})$ , and the condition number of  $A$  to be  $\kappa(A) := \lambda_{\max}(A)/\lambda_{\min}(A)$ . If we do not specify, the following  $\lambda_{\max}$ ,  $\lambda_{\min}$ , and  $\kappa$  are the eigenvalues and condition number of the connectivity matrix  $C = A^\top A$ . For any  $\mathbf{b} \in \mathbb{R}^m$ , we denote  $\mathbf{b}_A$  to be the projection of  $\mathbf{b}$  on the range space of  $A$ .

## 2.2 Optimization problems

In this subsection, we are going to introduce two optimization problems: *non-negative least squares* and  $\ell_1$  *minimization*.

### 2.2.1 Non-negative least squares

► **Problem 1** (non-negative least squares). Let  $m, n \in \mathbb{N}$ . Given  $A \in \mathbb{R}^{m \times n}$  and vector  $\mathbf{b} \in \mathbb{R}^m$ , find  $\mathbf{x} \in \mathbb{R}^n$  that minimizes  $\|\mathbf{b} - A\mathbf{x}\|_2^2/2$  subject to  $\mathbf{x}_i \geq 0$  for all  $i \in [n]$ .

► **Remark.** Recall that the least squares problem is defined as finding  $\mathbf{x}$  that minimize  $\|\mathbf{b} - A\mathbf{x}\|_2$ . That is, the non-negative least squares is a restricted version of the least squares problem. Nevertheless, one can use a non-negative least squares solver to solve the least squares problem by setting  $A' = \begin{pmatrix} A^\top A & -A^\top A \\ -A^\top A & A^\top A \end{pmatrix}$  and  $\mathbf{b}' = \begin{pmatrix} \mathbf{b} \\ -\mathbf{b} \end{pmatrix}$  where  $(A, \mathbf{b})$  is the instance of least squares and  $(A', \mathbf{b}')$  is the instance of non-negative least squares.

The SNN algorithm might not solve the non-negative least squares problem exactly and thus we define the following notion of solving the non-negative least squares problem *approximately*.

► **Definition 3** ( $\epsilon$ -approximate solution to non-negative least squares). Let  $m, n \in \mathbb{N}$  and  $\epsilon > 0$ . Given  $A \in \mathbb{R}^{m \times n}$  and  $\mathbf{b} \in \mathbb{R}^m$ . We say  $\mathbf{x}$  is an  $\epsilon$ -approximate solution to the non-negative least squares problem of  $(A, \mathbf{b})$  if  $\|A\mathbf{x} - A\mathbf{x}^*\|_2 \leq \epsilon \|\mathbf{b}\|_2$  where  $\mathbf{x}^*$  is an optimal solution.

### 2.2.2 $\ell_1$ minimization

► **Problem 2** ( $\ell_1$  minimization). Let  $m, n \in \mathbb{N}$ . Given  $A \in \mathbb{R}^{m \times n}$  and  $\mathbf{b} \in \mathbb{R}^m$  such that there exists a solution to  $A\mathbf{x} = \mathbf{b}$ . The goal of  $\ell_1$  minimization is to solve the following optimization problem.

$$\begin{aligned} & \underset{\mathbf{x} \in \mathbb{R}^n}{\text{minimize}} && \|\mathbf{x}\|_1 \\ & \text{subject to} && A\mathbf{x} = \mathbf{b}. \end{aligned}$$

Similarly, we do not expect SNN algorithm to solve the  $\ell_1$  minimization exactly. Thus, we define the notion of solving the  $\ell_1$  minimization problem *approximately* as follows.

► **Definition 4** ( $\epsilon$ -approximate solution to  $\ell_1$  minimization). Let  $m, n \in \mathbb{N}$  and  $\epsilon > 0$ . Given  $A \in \mathbb{R}^{m \times n}$  and  $\mathbf{b} \in \mathbb{R}^m$ . Let  $\mathbf{OPT}^{\ell_1}$  denote the optimal value of the  $\ell_1$  minimization problem of  $(A, \mathbf{b})$ . We say  $\mathbf{x} \in \mathbb{R}^n$  is an  $\epsilon$ -approximate solution of the  $\ell_1$  minimization problem of  $(A, \mathbf{b})$  if  $\|\mathbf{b} - A\mathbf{x}\|_2 \leq \epsilon \cdot \|\mathbf{b}\|_2$  and  $\|\mathbf{x}\|_1 - \mathbf{OPT}^{\ell_1} \leq \epsilon \cdot \mathbf{OPT}^{\ell_1}$ .

### 2.3 Karush-Kuhn-Tucker conditions

Karush-Kuhn-Tucker (KKT) conditions are necessary and sufficient conditions for the optimality of optimization problems under some regular assumptions. Consider the following optimization program.

$$\begin{aligned} & \underset{\mathbf{x} \in \mathbb{R}^n}{\text{minimize}} && f(\mathbf{x}) \\ & \text{subject to} && g_i(\mathbf{x}) \leq 0, \quad \forall i = 1, 2, \dots, m, \\ & && h_j(\mathbf{x}) = 0, \quad \forall j = 1, 2, \dots, k, \end{aligned} \tag{4}$$

where  $f, g_1, \dots, g_m, h_1, \dots, h_k$  are convex and differentiable. Let  $\mathbf{v} \in \mathbb{R}^m$  and  $\boldsymbol{\mu} \in \mathbb{R}^k$  be the dual variables. KKT conditions give necessary and sufficient conditions for  $(\mathbf{x}, \mathbf{v}, \boldsymbol{\mu})$  be a pair of primal and dual optimal solutions.

► **Theorem 5** (KKT conditions, Chapter 5.5.3 in [13]).  *$(\mathbf{x}, \mathbf{v}, \boldsymbol{\mu})$  are a pair of primal and dual optimal solutions for (4) if and only if the following conditions hold.*

- $\mathbf{x}$  is primal feasible, i.e.,  $g_i(\mathbf{x}) \leq 0$  and  $h_j(\mathbf{x}) = 0$  for all  $i \in [m]$  and  $j \in [k]$ .
- $(\mathbf{v}, \boldsymbol{\mu})$  is dual feasible, i.e.,  $\mathbf{v}_i \geq 0$  for all  $i \in [m]$ .
- The Lagrange multiplier vanishes, i.e.,  $\nabla f(\mathbf{x}) + \sum_{i \in [m]} \mathbf{v}_i \nabla g_i(\mathbf{x}) + \sum_{j \in [k]} \boldsymbol{\mu}_j \nabla h_j(\mathbf{x}) = 0$ .
- $(\mathbf{x}, \mathbf{v}, \boldsymbol{\mu})$  satisfy complementary slackness, i.e.,  $\mathbf{v}_i f_i(\mathbf{x}) \geq 0$  for all  $i \in [m]$ .

### 2.4 Perturbation theory

Perturbation theory, sometimes known as sensitivity analysis, for optimization problems concerns the situation where the optimization program is perturbed and the goal is to give a good estimation for the optimal solution. See a nice survey by Bonnans and Shapiro [11]. In the following we state a special case for convex optimization program with strong duality.

► **Theorem 6** (perturbation, Chapter 5.6 in [13]<sup>15</sup>). *Given the following two optimization programs where the strong duality holds and there exists feasible dual solution.*

$$\begin{aligned} & \underset{\mathbf{x}}{\text{minimize}} && f(\mathbf{x}) & & \underset{\mathbf{x}}{\text{minimize}} && f(\mathbf{x}) \\ & \text{subject to} && g_i(\mathbf{x}) \leq 0, \quad \forall i = 1, 2, \dots, m, & & \text{subject to} && g_i(\mathbf{x}) \leq \mathbf{a}_i, \quad \forall i = 1, 2, \dots, m, \\ & && h_j(\mathbf{x}) = 0, \quad \forall j = 1, 2, \dots, k. & & && h_j(\mathbf{x}) = \mathbf{b}_j, \quad \forall j = 1, 2, \dots, k. \end{aligned} \tag{5} \tag{6}$$

Let  $\text{OPT}^{\text{original}}$  be the optimal value of the original program (5) and  $\text{OPT}^{\text{perturbed}}$  be the optimal value of the perturbed program (6). Let  $(\mathbf{v}^*, \boldsymbol{\mu}^*) \in \mathbb{R}^m \times \mathbb{R}^k$  be the optimal dual solution of the perturbed program (6). We have

$$\text{OPT}^{\text{original}} \geq \text{OPT}^{\text{perturbed}} + \mathbf{a}^\top \mathbf{v}^* + \mathbf{b}^\top \boldsymbol{\mu}^*.$$

<sup>15</sup>Note that we switch the original and perturbed programs in the statement in [13].

### 3 A simple SNN algorithm for $\ell_1$ minimization

In this section, we focus on the discovery of the dual view of simple SNN and how it can be viewed as a *primal-dual algorithm* for solving the  $\ell_1$  minimization problem.

Recall that for the  $\ell_1$  minimization problem, we are working on the *two-sided* simple SNN for the convenience of future analysis. That is,

$$\frac{d}{dt} \mathbf{u}(t) = -\alpha \cdot A^\top A \mathbf{s}(t) + A^\top \mathbf{b},$$

where  $\mathbf{s}_i(t) = \mathbf{1}_{\mathbf{u}_i(t) > \eta} - \mathbf{1}_{\mathbf{u}_i(t) < -\eta}$ . To solve the  $\ell_1$  minimization problem, we configure a two-sided simple SNN as follows. Given an input  $(A, \mathbf{b})$ , let  $C = A^\top A$  and  $\mathbf{I} = A^\top \mathbf{b}$ . However, currently it is unclear how does the above simple SNN dynamics relate to the  $\ell_1$  minimization problem.

$$\begin{aligned} & \underset{\mathbf{x} \in \mathbb{R}^n}{\text{minimize}} && \|\mathbf{x}\|_1 \\ & \text{subject to} && A\mathbf{x} = \mathbf{b}. \end{aligned} \tag{7}$$

Interesting, the connection between simple SNN and the  $\ell_1$  minimization problem happens in the *dual program* of the  $\ell_1$  minimization problem. Before we formally explain this connection, let us write down the dual program of (7).

$$\begin{aligned} & \underset{\mathbf{v} \in \mathbb{R}^m}{\text{maximize}} && \mathbf{b}^\top \mathbf{v} \\ & \text{subject to} && \|A^\top \mathbf{v}\|_\infty \leq 1. \end{aligned} \tag{8}$$

Let us try to make some geometric observations on (8). First, the objective of the dual program is to maximize the inner product with  $\mathbf{b}$ , which is quite related to the external charging of SNN since we take  $\mathbf{I} = A^\top \mathbf{b}$ . Next, the feasible region of the dual program is a polytope (or a polyhedron) defined by the intersection of half-spaces  $\{\mathbf{v} \in \mathbb{R}^m : A_i^\top \mathbf{v} \leq 1\}$  and  $\{\mathbf{v} \in \mathbb{R}^m : -A_i^\top \mathbf{v} \leq 1\}$  for each  $i \in [n]$  where  $A_i$  denotes the  $i^{\text{th}}$  column of  $A$ .

It will be convenient to introduce the following notation before we move on. For  $i \in [n]$ , let  $A_{-i} = -A_i$ . Let  $[\pm n] = \{\pm 1, \pm 2, \dots, \pm n\}$ . Thus, the feasible polytope of the dual program is defined by the intersection of half-spaces defined by  $A_j^\top \mathbf{v} \leq 1$  for all  $j \in [\pm n]$ . We call this polytope the *dual polytope*<sup>16</sup>. Moreover, for each  $j \in [\pm n]$ , we call the hyperplane  $\{\mathbf{v} : A_j^\top \mathbf{v} = 1\}$  the *wall*  $W_j$  of the dual polytope. See Figure 3 for examples.

Now, the key observation is that by a linear transformation, the dynamics of simple SNN has a natural interpretation in the dual space. We call it the *dual SNN* defined as follows.

#### 3.1 Dual SNN

We first recall the simple SNN dynamics which we call the *primal SNN* from now on. For convenience, we set the threshold parameter  $\eta = 1$ . For any  $t \geq 0$ ,

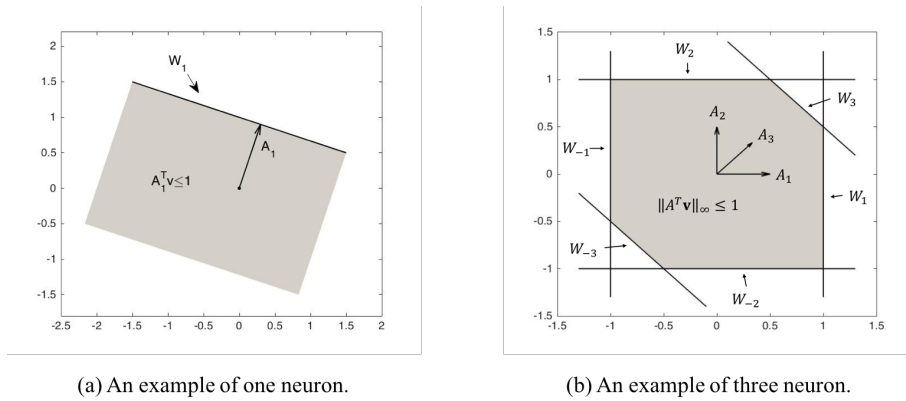
$$\mathbf{u}(t + dt) = \mathbf{u}(t) - \alpha \cdot A^\top A \cdot \mathbf{s}(t) + A^\top \mathbf{b} \cdot dt. \tag{9}$$

Now, we define the dual SNN  $\mathbf{v}(t) \in \mathbb{R}^m$  as follows. Let  $\mathbf{v}(0) = \mathbf{0}$  and for each  $t \geq 0$ , define

$$\mathbf{v}(t + dt) = \mathbf{v}(t) - \alpha \cdot A \mathbf{s}(t) + \mathbf{b} \cdot dt. \tag{10}$$

<sup>16</sup>In the case where the feasible region of the dual program is not bounded, it is a dual *polyhedron*. For the convenience of the presentation, we usually assume the feasible region is bounded.





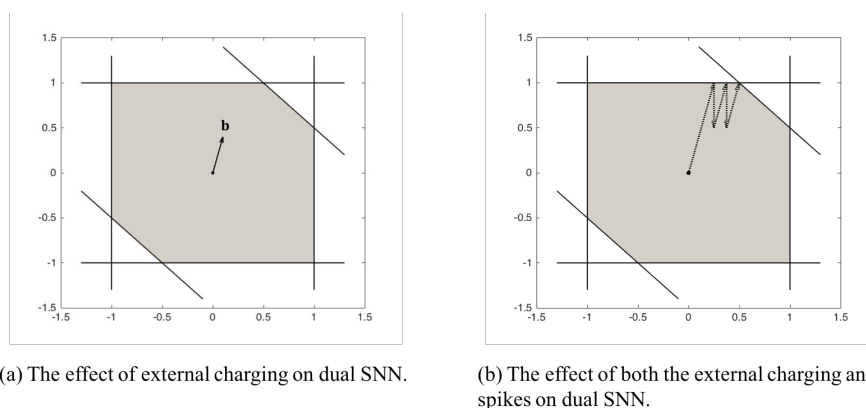
■ **Figure 3** This is examples of the geometric interpretation of the dual program of  $\ell_1$  minimization problem. In (a), we have  $n = 1$  where  $A_1 = [\frac{1}{3} \ 1]^T$ . In this case, the gray area, i.e., the feasible region of the dual program, is unbounded. In (b), we have  $n = 3$  where  $A_1 = [1 \ 0]^T$ ,  $A_2 = [0 \ 1]^T$ , and  $A_3 = [\frac{2}{3} \ \frac{2}{3}]^T$ . In this case, the gray area is bounded and thus called dual polytope.

Let us make some remarks about the connection between the primal and dual SNNs. First, it can be immediately seen that  $\mathbf{u}(t) = A^T \mathbf{v}(t)$  for each  $t \in \mathbb{N}$  from (9) and (10). That is, given  $\mathbf{v}(t)$ , it is easy to get  $\mathbf{u}(t)$  by multiplying  $\mathbf{u}(t)$  with  $A^T$  on the left. It turns out that the other direction also holds. For each  $t \in \mathbb{N}$ , we have  $\mathbf{v}(t) = (A^T)^\dagger \mathbf{u}(t)$ , where  $(A^T)^\dagger$  is the pseudo-inverse of  $A^T$ . The reason is because the primal SNN  $\mathbf{u}(t)$  lies in the column space of  $A$ . Thus, the two dynamics are in fact *isomorphic* to each other.

Now let us understand the dynamics of dual SNN in the dual space  $\mathbb{R}^m$ . At each timestep, there are two terms, i.e., the external charging  $\mathbf{b} \cdot dt$  and the spiking effect  $-\alpha \mathbf{A} \mathbf{s}(t)$ , that affect the dual SNN  $\mathbf{v}(t)$ . The external charging can be thought of as a constant force that drags that dual SNN in the direction  $\mathbf{b}$ . See Figure 4. This coincides with the objective function of the dual program (8) and thus the external charging can then be viewed as taking a *gradient* step towards solving (8).

Nevertheless, to solve (8), one need to make sure the solution  $\mathbf{v}$  is feasible, i.e.,  $\mathbf{v}$  should lie in the dual polytope. Interestingly, this is exactly what the spike is doing! Recall that neuron  $i$  fires a spike if  $|u_i(t)| > 1$  (recall that we set  $\eta = 1$ ), which corresponds to  $|A_i^T \mathbf{v}(t)| > 1$  in the dual space. Thus, the spike term has the following nice geometric interpretation: if  $\mathbf{v}(t)$  “exceeds” the wall  $W_j$  for some  $j \in [\pm n]$ , then neuron  $|j|$  fires a spike and  $\mathbf{v}(t)$  is “bounced back” in the normal direction of the wall  $W_j$  in the sense that  $\mathbf{v}(t)$  is subtracted by  $\alpha \cdot A_j$ . See Figure 4 for example.

Therefore, one can view the dual SNN as performing a variant of projected gradient descent algorithm for the dual program of  $\ell_1$  minimization problem. Specifically, to maintain the feasibility, the vector is not projected back to the feasible region as usual, but is “bounced back” in the normal direction of the wall  $W_j$  corresponding to the violated constraint  $A_j^T \mathbf{v} \leq 1$ . An advantage of this variant is that the “bounced back” operation is simply subtraction of  $\alpha \cdot A_j$ , which is significantly more efficient than the orthogonal projection back to the feasible region. On the other hand, note that the dynamics might not exactly converge to the optimal dual solution  $\mathbf{v}^{\text{OPT}}$ . Intuitively, the best we can hope for is that  $\mathbf{v}(t)$  will converge to a small neighboring region of  $\mathbf{v}^{\text{OPT}}$  (assuming the spiking strength  $\alpha$  is sufficiently small). The above intuition of viewing dual SNN as a projected gradient descent algorithm for the dual program of the  $\ell_1$ -minimization problem will be formally proved in the full version of this paper.



■ **Figure 4** This is examples of the geometric interpretation of the dual We consider the same matrix  $A$  as in Figure 3 and  $\mathbf{b} = [0.1 \ 0.4]^T$ . In (a), one can see that the external charging  $\mathbf{b}$  points the direction that dual SNN is moving. In (b), one can see that the effect of spikes on dual SNN is a jump in the direction of the normal vector of the wall.

**The primal-dual connection.** So far we have informally seen that the dual SNN can be viewed as solving the dual program of the  $\ell_1$ -minimization problem. However, this does not immediately give us a reason why the firing rate would converge to the solution of the primal program. It turns out that there is a beautiful connection between the dual SNN and firing rate through the *Karush-Kuhn-Tucker (KKT) conditions* (see Section 2.3) and perturbation theory (see Section 2.4).

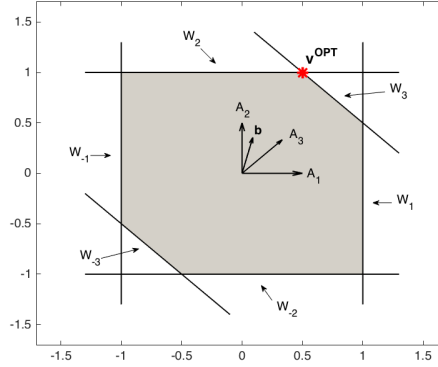
We now discuss some intuitions about how the dual solution translates to the primal solution. To jump into the core idea, let us consider an ideal scenario where the dual SNN  $\mathbf{v}(t)$  is already very close to the optimal dual solution  $\mathbf{v}^{\text{OPT}}$  for the dual program of the  $\ell_1$  minimization problem. Since  $\mathbf{v}^{\text{OPT}}$  is the optimal solution and thus it must lie on the boundary of the dual polytope. Let  $\Gamma \subseteq [\pm n]$  be the set of walls that  $\mathbf{v}^{\text{OPT}}$  touches. That is,  $j \in \Gamma$  if and only if  $A_j^T \mathbf{v}^{\text{OPT}} = 1$ . Now, let  $\mathbf{x}^{\text{OPT}}$  denote the optimal primal solution of the  $\ell_1$  minimization problem. Observe that by the complementary slackness in the KKT conditions, for each  $i \in [n]$ , we have  $\mathbf{x}_i^{\text{OPT}} > 0$  (resp.  $\mathbf{x}_i^{\text{OPT}} < 0$ ) if  $i \in \Gamma$  (resp.  $-i \in \Gamma$ ) and  $\mathbf{x}_i^{\text{OPT}} = 0$  if  $i, -i \notin \Gamma$ . To summary, this is saying that  $\Gamma$  contains the coordinates that are non-zero in the primal optimal solution  $\mathbf{x}^{\text{OPT}}$ . See Figure 5 for an example.

With this observation, once the dual SNN  $\mathbf{v}(t)$  is very close to the optimal dual solution  $\mathbf{v}^{\text{OPT}}$  and stays nearby, only those neurons correspond to  $\Gamma$  would fire spikes. In other words, the firing rate of the non-zero coordinates in the primal optimal solution  $\mathbf{x}^{\text{OPT}}$  will remain non-zero due to the spikes while the other coordinates will gradually go to zero.

At this point, we have seen that (i) the dual SNN can be viewed as a projected gradient descent algorithm for the dual program of  $\ell_1$  minimization problem and (ii) the dual solution (resp. dual SNN) and primal solution (resp. firing rate) have a natural connection through the KKT conditions. Now, let us formally state the main theorem of this section about simple SNN solving  $\ell_1$  minimization problem.

► **Theorem 7.** *Given  $A \in \mathbb{R}^{m \times n}$  and  $\mathbf{b} \in \mathbb{R}^m$  where all the row of  $A$  has unit norm. Let  $\gamma(A)$  be the niceness parameter<sup>17</sup> of  $A$ . Suppose  $\gamma(A) > 0$  and there exists a solution to  $A\mathbf{x} = \mathbf{b}$ . There exists a polynomial  $\alpha(\cdot)$  such that for any  $t \geq 0$ , let  $\mathbf{x}(t)$  be the firing rate*

<sup>17</sup>The niceness parameter is formally defined in Definition 4 of the full version of this paper.



■ **Figure 5** This is an example based on Figure 3 and Figure 4. In this example,  $A_1 = [1 \ 0]^\top$ ,  $A_2 = [0 \ 1]^\top$ ,  $A_3 = [\frac{2}{3} \ \frac{2}{3}]^\top$ , and  $\mathbf{b} = [0.1 \ 0.4]^\top$ . The optimal dual solution is  $\mathbf{v}^{\text{OPT}} = [\frac{1}{2} \ 1]^\top$  as shown in the figure. Thus, by the above definition we have  $\Gamma = \{2, 3\}$ . By the KKT conditions, we then know that only the 2<sup>nd</sup> and 3<sup>rd</sup> coordinate of the optimal primal solution are non-zero. Indeed, the optimal primal solution is  $\mathbf{x}^{\text{OPT}} = [0 \ \frac{3}{10} \ \frac{3}{20}]^\top$ .

of the simple SNN with  $C = A^\top A$ ,  $\mathbf{I} = A^\top \mathbf{b}$ ,  $\eta = 1$ ,  $0 < \alpha \leq \alpha(\frac{\gamma(A)}{n \cdot \lambda_{\max}})$ . Let  $\text{OPT}^{\ell_1}$  be the optimal value of the  $\ell_1$  minimization problem. For any  $\epsilon > 0$ , when  $t \geq \Omega(\frac{m^2 \cdot n \cdot \|\mathbf{b}\|_2^2}{\epsilon^2 \cdot \lambda_{\min} \cdot \text{OPT}^{\ell_1}})$ , then  $\mathbf{x}(t)$  is an  $\epsilon$ -approximate solution to the  $\ell_1$  minimization problem for  $(A, \mathbf{b})$ .

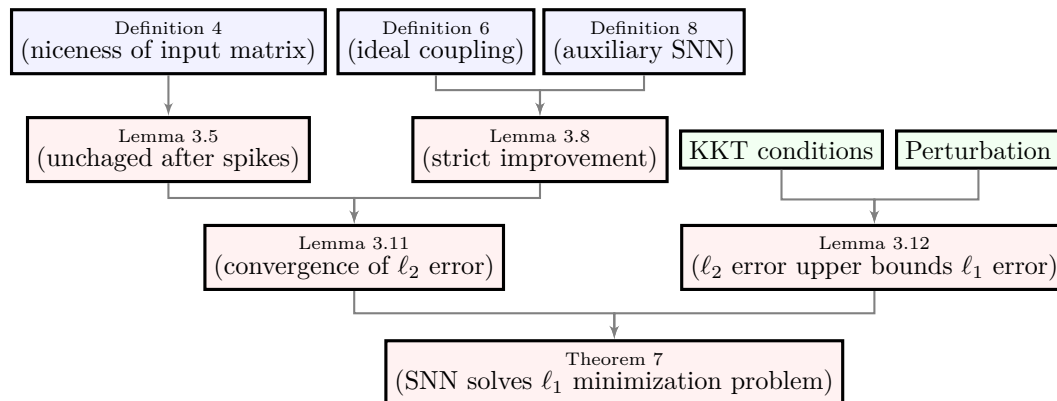
Two remarks on the statement of Theorem 7. First, we consider the *continuous SNN* instead of the discrete SNN, which is of interest for simulation on classical computer. In discrete SNN, the *step size* is some non-negligible  $\Delta t > 0$  instead of  $dt$ . The main reason for considering continuous SNN is that this significantly simplify the proof by avoiding a huge amount of nasty calculations. We suspect that the proof idea would hold for discrete SNN with discretization parameter  $\Delta t \leq \Delta t(\frac{\gamma(A)}{n \cdot \lambda_{\max}})$  for some polynomial  $\Delta t(\cdot)$ . Second, the parameters in Theorem 7 have not been optimized and we believe all the dependencies can be improved. Since the parameters highly affect the efficiency of SNN as an algorithm for  $\ell_1$  minimization problem, we pose it as an interesting open problem to study what are the best dependencies one can get.

### 3.2 Overview of the proof for Theorem 7

The proof for Theorem 7 consists of two main steps as mentioned in the previous subsection. The first step argues that the dual SNN  $\mathbf{v}(t)$  would converge to the neighborhood of the optimal dual solution  $\mathbf{v}^{\text{OPT}}$ . The second step is connecting the dual solution (i.e., the dual SNN) to the primal solution (i.e., the firing rate). In the subsection, we sketch the proof for Theorem 7 while some lemmas and definitions might not appear in this conference version and can found in the full version of this paper.

In the first step, we try to identify a *potential function*<sup>18</sup> that captures how close is  $\mathbf{v}(t)$  to the optimal dual solution  $\mathbf{v}^{\text{OPT}}$ . It turns out that this is not an easy task since the effect of spikes makes the behavior of dual SNN very non-monotone. We conquer the difficulty via a technique that we call *ideal coupling* (see Definition 6 and Figure 7 in the full version).

<sup>18</sup> Potential function is widely used in the analysis of many gradient-descent based algorithm. The difficulty lies in the search of a good potential function for the algorithm.



■ **Figure 6** Overview of the proof for Theorem 7. The missing lemmas and definitions can be found in the full version of this paper.

The idea is associating the dual SNN  $\mathbf{v}(t)$  with an *ideal SNN*  $\mathbf{v}^{\text{ideal}}(t)$  for every  $t \geq 0$  such that the ideal SNN would have *smoother* behavior comparing to the spiking phenomenon in the dual SNN. We will formally define the ideal SNN in Section 3.4 of the full version. There are two advantages of using ideal SNN instead of handling dual SNN directly: (i) Ideal SNN is smoother than dual SNN in the sense that it would not change after spikes (see Lemma 3.5 in the full version). Further, by introducing some auxiliary processes (i.e., the auxiliary SNNs defined in Definition 8 in the full version), we are able to identify a potential function that is strictly improving at any moment and measures how well the dual SNN has been solving the  $\ell_1$  minimization problem (see Lemma 3.8 in the full version). (ii) ideal SNN is naturally associated with an *ideal solution* (defined in Definition 7 in the full version) which is easier to analyze than the firing rate. Using these good properties of ideal SNN, we can prove in Lemma 3.11 (in the full version) that the  $\ell_2$  residual error of the ideal solution will converge to 0.

After we are able to show the convergence of the  $\ell_2$  residual error in Lemma 3.11 (in the full version), we move to the second step where the goal is showing that the  $\ell_1$  norm of the solution is also small. We look at the KKT conditions of the  $\ell_1$  minimization problem and observe that the primal and dual solutions of SNN satisfy the KKT conditions of a *perturbed* program of the  $\ell_1$  minimization problem. Finally, combine tools from perturbation theory, we can upper bound the  $\ell_1$  error of the ideal solution by its  $\ell_2$  residual error in Lemma 3.12 (in the full version).

Theorem 7 then follows from Lemma 3.11 and Lemma 3.12 (in the full version) with some special cares on how to transform everything for ideal solution to the firing rate. See Figure 6 for an overall structure of the proof for Theorem 7.

The full proof for Theorem 7 and other technical details are all provided in the full version of this paper.

## References

- 1 Moshe Abeles. *Corticonics: Neural circuits of the cerebral cortex*. Cambridge University Press, 1991.
- 2 Christina Allen and Charles F Stevens. An evaluation of causes for unreliability of synaptic transmission. *Proceedings of the National Academy of Sciences*, 91(22):10380–10383, 1994.
- 3 Arunava Banerjee. Learning Precise Spike Train-to-Spike Train Transformations in Multilayer Feedforward Neuronal Networks. *Neural computation*, 28(5):826–848, 2016.

- 4 David G Barrett, Sophie Denève, and Christian K Machens. Firing rate predictions in optimal balanced networks. In *Advances in Neural Information Processing Systems*, pages 1538–1546, 2013.
- 5 Amir Beck and Marc Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM journal on imaging sciences*, 2(1):183–202, 2009.
- 6 Yoshua Bengio, Thomas Mesnard, Asja Fischer, Saizheng Zhang, and Yuhuai Wu. STDP-compatible approximation of backpropagation in an energy-based model. *Neural computation*, 29(3):555–577, 2017.
- 7 William Bialek, Fred Rieke, RR De Ruyter Van Steveninck, and David Warland. Reading a neural code. *Science*, 252(5014):1854–1857, 1991.
- 8 Jonathan Binas, Giacomo Indiveri, and Michael Pfeiffer. Spiking Analog VLSI Neuron Assemblies as Constraint Satisfaction Problem Solvers. *arXiv preprint arXiv:1511.00540*, 2015.
- 9 Anmol Biswas, Sidharth Prasad, Sandip Lashkare, and Udayan Ganguly. A simple and efficient SNN and its performance & robustness evaluation method to enable hardware implementation. *arXiv preprint arXiv:1612.02233*, 2016.
- 10 Vincenzo Bonifaci, Kurt Mehlhorn, and Girish Varma. Physarum can compute shortest paths. *Journal of Theoretical Biology*, 309:121–133, 2012.
- 11 J Frédéric Bonnans and Alexander Shapiro. Optimization problems with perturbations: A guided tour. *SIAM review*, 40(2):228–264, 1998.
- 12 Olaf Booij and Hieu tat Nguyen. A gradient descent rule for spiking neurons emitting multiple spikes. *Information Processing Letters*, 95(6):552–558, 2005.
- 13 Stephen Boyd and Lieven Vandenberghe. *Convex optimization*. Cambridge university press, 2004.
- 14 Nicolas Brunel and Peter E Latham. Firing rate of the noisy quadratic integrate-and-fire neuron. *Neural Computation*, 15(10):2281–2306, 2003.
- 15 Lars Buesing, Johannes Bill, Bernhard Nessler, and Wolfgang Maass. Neural dynamics as sampling: a model for stochastic computation in recurrent networks of spiking neurons. *PLoS Comput Biol*, 7(11):e1002211, 2011.
- 16 Bernard Chazelle. Natural algorithms. In *Proceedings of the twentieth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 422–431. Society for Industrial and Applied Mathematics, 2009.
- 17 Bernard Chazelle. Natural algorithms and influence systems. *Communications of the ACM*, 55(12):101–110, 2012.
- 18 Scott Shaobing Chen, David L Donoho, and Michael A Saunders. Atomic decomposition by basis pursuit. *SIAM review*, 43(1):129–159, 2001.
- 19 Peter U Diehl and Matthew Cook. Unsupervised learning of digit recognition using spike-timing-dependent plasticity. *Frontiers in computational neuroscience*, 9:99, 2015.
- 20 A Aldo Faisal, Luc PJ Selen, and Daniel M Wolpert. Noise in the nervous system. *Nature reviews neuroscience*, 9(4):292–303, 2008.
- 21 Richard FitzHugh. Impulses and physiological states in theoretical models of nerve membrane. *Biophysical journal*, 1(6):445–466, 1961.
- 22 Nicolas Fourcaud-Trocmé, David Hansel, Carl Van Vreeswijk, and Nicolas Brunel. How spike generation mechanisms determine the neuronal response to fluctuating inputs. *Journal of Neuroscience*, 23(37):11628–11640, 2003.
- 23 Wulfram Gerstner. Time structure of the activity in neural network models. *Physical review E*, 51(1):738, 1995.
- 24 Tim Gollisch and Markus Meister. Rapid neural coding in the retina with relative spike latencies. *science*, 319(5866):1108–1111, 2008.
- 25 Walter Heiligenberg. *Neural nets in electric fish*. MIT press Cambridge, MA, 1991.

- 26 James L Hindmarsh and RM Rose. A model of neuronal bursting using three coupled first order differential equations. *Proc. R. Soc. Lond. B*, 221(1222):87–102, 1984.
- 27 Alan L Hodgkin and Andrew F Huxley. A quantitative description of membrane current and its application to conduction and excitation in nerve. *The Journal of physiology*, 117(4):500, 1952.
- 28 John J Hopfield. Pattern recognition computation using action potential timing for stimulus representation. *Nature*, 376(6535):33, 1995.
- 29 Eugene M Izhikevich et al. Simple model of spiking neurons. *IEEE Transactions on neural networks*, 14(6):1569–1572, 2003.
- 30 Zeno Jonke, Stefan Habenschuss, and Wolfgang Maass. A theoretical basis for efficient computations with noisy spiking neurons. *arXiv preprint arXiv:1412.5862*, 2014.
- 31 Zeno Jonke, Stefan Habenschuss, and Wolfgang Maass. Solving constraint satisfaction problems with networks of spiking neurons. *Frontiers in neuroscience*, 10, 2016.
- 32 Saeed Reza Kheradpisheh, Mohammad Ganjtabesh, and Timothée Masquelier. Bio-inspired unsupervised learning of visual features leads to robust invariant object recognition. *Neuro-computing*, 205:382–392, 2016.
- 33 Werner M Kistler, Wulfram Gerstner, and J Leo van Hemmen. Reduction of the Hodgkin-Huxley equations to a single-variable threshold model. *Neural computation*, 9(5):1015–1045, 1997.
- 34 Nobuyuki Kuwabara and Nobuo Suga. Delay lines and amplitude selectivity are created in subthalamic auditory nuclei: the brachium of the inferior colliculus of the mustached bat. *Journal of neurophysiology*, 69(5):1713–1724, 1993.
- 35 Louis Lapicque. Recherches quantitatives sur l’excitation électrique des nerfs traitée comme une polarisation. *J. Physiol. Pathol. Gen*, 9(1):620–635, 1907.
- 36 Tsung-Han Lin and Ping Tak Peter Tang. Dictionary Learning by Dynamical Neural Networks. *arXiv preprint arXiv:1805.08952*, 2018.
- 37 Adi Livnat and Christos Papadimitriou. Sex as an algorithm: the theory of evolution under the lens of computation. *Communications of the ACM*, 59(11):84–93, 2016.
- 38 Adi Livnat, Christos Papadimitriou, Aviad Rubinstein, Gregory Valiant, and Andrew Wan. Satisfiability and evolution. In *Foundations of Computer Science (FOCS), 2014 IEEE 55th Annual Symposium on*, pages 524–530. IEEE, 2014.
- 39 Nancy Lynch and Cameron Musco. A Basic Compositional Model for Spiking Neural Networks. *arXiv preprint arXiv:1808.03884*, 2018.
- 40 Nancy Lynch, Cameron Musco, and Merav Parter. Spiking Neural Networks: An Algorithmic Perspective. In *Workshop on Biological Distributed Algorithms (BDA), July 28th, 2017, Washington DC, USA*, 2017.
- 41 Nancy A. Lynch, Cameron Musco, and Merav Parter. Computational Tradeoffs in Biological Neural Networks: Self-Stabilizing Winner-Take-All Networks. In *8th Innovations in Theoretical Computer Science Conference, ITCS 2017, January 9-11, 2017, Berkeley, CA, USA*, pages 15:1–15:44, 2017. doi:10.4230/LIPIcs.ITCS.2017.15.
- 42 Nancy A. Lynch, Cameron Musco, and Merav Parter. Neuro-RAM Unit with Applications to Similarity Testing and Compression in Spiking Neural Networks. In *31st International Symposium on Distributed Computing, DISC 2017, October 16-20, 2017, Vienna, Austria*, pages 33:1–33:16, 2017. doi:10.4230/LIPIcs.DISC.2017.33.
- 43 Wolfgang Maass. Lower bounds for the computational power of networks of spiking neurons. *Neural computation*, 8(1):1–40, 1996.
- 44 Wolfgang Maass. Fast sigmoidal networks via spiking neurons. *Neural Computation*, 9(2):279–304, 1997.
- 45 Wolfgang Maass. Networks of spiking neurons: the third generation of neural network models. *Neural networks*, 10(9):1659–1671, 1997.

- 46 Wolfgang Maass. Computing with spiking neurons. *Pulsed neural networks*, 85, 1999.
- 47 Wolfgang Maass. To Spike or Not to Spike: That Is the Question. *Proceedings of the IEEE*, 103(12):2219–2224, 2015.
- 48 Wolfgang Maass and Christopher M Bishop. *Pulsed neural networks*. MIT press, 2001.
- 49 Catherine Morris and Harold Lecar. Voltage oscillations in the barnacle giant muscle fiber. *Biophysical journal*, 35(1):193–213, 1981.
- 50 Hesham Mostafa, Lorenz K Müller, and Giacomo Indiveri. An event-based architecture for solving constraint satisfaction problems. *Nature communications*, 6, 2015.
- 51 Toshiyuki Nakagaki, Hiroyasu Yamada, and Ágota Tóth. Intelligence: Maze-solving by an amoeboid organism. *Nature*, 407(6803):470–470, 2000.
- 52 Bruno A Olshausen and David J Field. Emergence of simple-cell receptive field properties by learning a sparse code for natural images. *Nature*, 381(6583):607, 1996.
- 53 Hélene Paugam-Moisy and Sander Bohte. Computing with spiking neuron networks. In *Handbook of natural computing*, pages 335–376. Springer, 2012.
- 54 Fred Rieke and David Warland. *Spikes: exploring the neural code*. MIT press, 1999.
- 55 Christopher J Rozell, Don H Johnson, Richard G Baraniuk, and Bruno A Olshausen. Sparse coding via thresholding and local competition in neural circuits. *Neural computation*, 20(10):2526–2563, 2008.
- 56 Rufin Van Rullen and Simon J Thorpe. Rate coding versus temporal order coding: what the retinal ganglion cells tell the visual cortex. *Neural computation*, 13(6):1255–1283, 2001.
- 57 Michael N Shadlen and William T Newsome. Noise, neural codes and cortical organization. *Current opinion in neurobiology*, 4(4):569–579, 1994.
- 58 Samuel Shapero, Christopher Rozell, and Paul Hasler. Configurable hardware integrate and fire neurons for sparse approximation. *Neural Networks*, 45:134–143, 2013.
- 59 Samuel Shapero, Mengchen Zhu, Jennifer Hasler, and Christopher Rozell. Optimal sparse approximation with integrate and fire neurons. *International journal of neural systems*, 24(05):1440001, 2014.
- 60 Sumit Bam Shrestha and Qing Song. Robust learning in SpikeProp. *Neural Networks*, 86:54–68, 2017.
- 61 Richard B Stein. A theoretical analysis of neuronal variability. *Biophysical Journal*, 5(2):173, 1965.
- 62 Ping Tak Peter Tang. Convergence of LCA Flows to (C) LASSO Solutions. *arXiv preprint arXiv:1603.01644*, 2016.
- 63 Ping Tak Peter Tang, Tsung-Han Lin, and Mike Davies. Sparse Coding by Spiking Neural Networks: Convergence Theory and Computational Results. *arXiv preprint arXiv:1705.05475*, 2017.
- 64 Wondimu Teka, Toma M Marinov, and Fidel Santamaria. Neuronal spike timing adaptation described with a fractional leaky integrate-and-fire model. *PLoS computational biology*, 10(3):e1003526, 2014.
- 65 Atsushi Tero, Ryo Kobayashi, and Toshiyuki Nakagaki. A mathematical model for adaptive transport network in path finding by true slime mold. *Journal of theoretical biology*, 244(4):553–564, 2007.
- 66 Simon Thorpe, Arnaud Delorme, and Rufin Van Rullen. Spike-based strategies for rapid processing. *Neural networks*, 14(6-7):715–725, 2001.
- 67 Simon Thorpe, Denis Fize, and Catherine Marlot. Speed of processing in the human visual system. *nature*, 381(6582):520, 1996.
- 68 Joel Zylberberg, Jason Timothy Murphy, and Michael Robert DeWeese. A sparse coding model with synaptically local plasticity and spiking neurons can account for the diverse shapes of V1 simple cell receptive fields. *PLoS computational biology*, 7(10):e1002250, 2011.



# Last-Iterate Convergence: Zero-Sum Games and Constrained Min-Max Optimization

Constantinos Daskalakis<sup>1</sup>

CSAIL, MIT, Cambridge MA, USA  
costis@csail.mit.edu

Ioannis Panageas<sup>2</sup>

ISTD, SUTD, Singapore  
ioannis@sutd.edu.sg

---

## Abstract

Motivated by applications in Game Theory, Optimization, and Generative Adversarial Networks, recent work of Daskalakis et al [Daskalakis et al., ICLR, 2018] and follow-up work of Liang and Stokes [Liang and Stokes, 2018] have established that a variant of the widely used Gradient Descent/Ascent procedure, called “Optimistic Gradient Descent/Ascent (OGDA)”, exhibits last-iterate convergence to saddle points in *unconstrained* convex-concave min-max optimization problems. We show that the same holds true in the more general problem of *constrained* min-max optimization under a variant of the no-regret Multiplicative-Weights-Update method called “Optimistic Multiplicative-Weights Update (OMWU)”. This answers an open question of Syrgkanis et al [Syrgkanis et al., NIPS, 2015].

The proof of our result requires fundamentally different techniques from those that exist in no-regret learning literature and the aforementioned papers. We show that OMWU monotonically improves the Kullback-Leibler divergence of the current iterate to the (appropriately normalized) min-max solution until it enters a neighborhood of the solution. Inside that neighborhood we show that OMWU becomes a contracting map converging to the exact solution. We believe that our techniques will be useful in the analysis of the last iterate of other learning algorithms.

**2012 ACM Subject Classification** Theory of computation → Convergence and learning in games

**Keywords and phrases** No regret learning, Zero-sum games, Convergence, Dynamical Systems, KL divergence

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2019.27

**Acknowledgements** Part of this work was done while the authors were visiting Northwestern University for a spring program in Econometrics.

## 1 Introduction

A central problem in Game Theory and Optimization is computing a pair of probability vectors  $(\mathbf{x}, \mathbf{y})$ , solving

$$\min_{\mathbf{y} \in \Delta_m} \max_{\mathbf{x} \in \Delta_n} \mathbf{x}^\top \mathbf{A} \mathbf{y}, \tag{1}$$

---

<sup>1</sup> Supported by NSF awards CCF-1617730 and IIS-1741137, a Simons Investigator Award, a Google Faculty Research Award, and an MIT-IBM Watson AI Lab research grant.

<sup>2</sup> Supported by SRG ISTD 2018 136. Part of this work was done while Ioannis was postdoc at MIT.



## 27:2 Last-Iterate Convergence

where  $\Delta_n \subset \mathbb{R}^n$  and  $\Delta_m \subset \mathbb{R}^m$  are probability simplices, and  $A$  is a  $n \times m$  matrix. Von Neumann's celebrated minimax theorem informs us that

$$\min_{\mathbf{y} \in \Delta_m} \max_{\mathbf{x} \in \Delta_n} \mathbf{x}^\top A \mathbf{y} = \max_{\mathbf{x} \in \Delta_n} \min_{\mathbf{y} \in \Delta_m} \mathbf{x}^\top A \mathbf{y}, \quad (2)$$

and that all solutions to the LHS are solutions to the RHS, and vice versa. This result was a founding stone in the development of Game Theory. Indeed, interpreting  $\mathbf{x}^\top A \mathbf{y}$  as the payment of the “min player” to the “max player” when the former selects a distribution  $\mathbf{y}$  over columns and the latter selects a distribution  $\mathbf{x}$  over rows of matrix  $A$ , a solution to (1) constitutes an equilibrium of the game defined by matrix  $A$ , called a “minimax equilibrium”, a pair of randomized strategies such that neither player can improve their payoff by unilaterally changing their distribution.

Besides their fundamental value for Game Theory, it is known that (1) and (2) are also intimately related to Linear Programming. It was shown by von Neumann that (2) follows from strong linear programming duality. Moreover, it was suggested by Dantzig [7] and recently proven by Adler [1] that any linear program can be solved by solving some min-max problem of the form (1). In particular, min-max problems of form (1) are exactly as expressive as min-max problems of the following form, which capture any linear program (by Lagrangifying the constraints):

$$\min_{\mathbf{y} \geq 0} \max_{\mathbf{x} \geq 0} (\mathbf{x}^\top A \mathbf{y} + b^\top \mathbf{x} + c^\top \mathbf{y}). \quad (3)$$

Soon after the minimax theorem was proven and its connection to linear programming was forged, researchers proposed dynamics for solving min-max optimization problems by having the min and max players of (1) run a simple learning procedure in tandem. An early method, proposed by Brown [4] and analyzed by Robinson [18], was fictitious play. Soon after, Blackwell's approachability theorem [3] propelled the field of online learning, which led to the discovery of several learning algorithms converging to minimax equilibrium at faster rates, while also being robust to adversarial environments, situations where one of the players of the game deviates from the prescribed dynamics; see e.g. [5]. These learning methods, called “no-regret”, include the celebrated multiplicative-weights-update method, follow-the-regularized-leader, and follow-the-perturbed-leader. Compared to centralized linear programming procedures the advantage of these methods is the simplicity of executing their steps, and their robustness to adversarial environments, as we just discussed.

### Last vs Average Iterate Convergence

Despite the extensive literature on no-regret learning, an unsatisfactory feature of known results is that min-max equilibrium is shown to be attained only in an average sense. To be precise, if  $(\mathbf{x}^t, \mathbf{y}^t)$  is the trajectory of a no-regret learning method, it is usually shown that the average  $\frac{1}{t} \sum_{\tau \leq t} \mathbf{x}^\tau \mathbf{y}^\tau$  converges to the optimal value of (1), as  $t \rightarrow \infty$ . Moreover, if the solution to (1) is unique, then  $\frac{1}{t} \sum_{\tau \leq t} (\mathbf{x}^\tau, \mathbf{y}^\tau)$  converges to the optimal solution. Unfortunately that does not mean that the last iterate  $(\mathbf{x}^t, \mathbf{y}^t)$  converges to an optimal solution, and indeed it commonly diverges or enters a limit cycle. Furthermore, in the optimization literature, Nesterov [15] provides a method that can give pointwise convergence (i.e., convergence of the last iterate) to problem (1)<sup>3</sup>, however his algorithm is not a no-regret

<sup>3</sup> Nesterov showed that by optimizing  $f_\mu(\mathbf{x}) := \mu \ln(\frac{1}{m} \sum_{j=1}^m e^{-\frac{1}{\mu}(A\mathbf{x})_j})$ ,  $g_\nu(\mathbf{y}) := \nu \ln(\frac{1}{n} \sum_{j=1}^n e^{\frac{1}{\nu}(A^\top \mathbf{y})_j})$  for  $\mu = \Theta(\frac{\epsilon}{\log m})$ ,  $\nu = \Theta(\frac{\epsilon}{\log n})$  yields an  $O(\epsilon)$  approximation to the problem (1).

learning algorithm. Recent work by Daskalakis et al [8] and Liang and Stokes [11] studies whether last iterate convergence can be established for no-regret learning methods in the simple unconstrained min-max problem of the form:

$$\min_{\mathbf{y} \in \mathbb{R}^m} \max_{\mathbf{x} \in \mathbb{R}^n} (\mathbf{x}^\top A \mathbf{y} + b^\top \mathbf{x} + c^\top \mathbf{y}). \quad (4)$$

For this problem, it is known that Gradient Descent/Ascent (GDA) is a no-regret learning procedure, corresponding to follow-the-regularized leader (FTRL) with  $\ell_2^2$ -regularization. As such, the average trajectory traveled by GDA converges to a min-max solution, in the afore-described sense. On the other hand, it is also known that GDA may diverge from the min-max solution, even in trivial cases such as  $A = I, n = m = 1, b = c = 0$ . Interestingly, [8, 11] show that a variant of GDA, called “Optimistic Gradient Descent/Ascent (OGDA),<sup>4</sup> exhibits last iterate convergence. Inspired by their theoretical result for the performance of OGDA in (4), Daskalakis et al. [8] even propose the use of OGDA for training Generative Adversarial Networks (GANs) [10]. Moreover, Syrgkanis et al. [19] provide numerical experiments which indicate that the trajectories of Optimistic Hedge (variant of Hedge in the same way OGDA is a variant of GDA) stabilize (i.e., converge pointwise) as opposed to (classic) Hedge and they *posed the question whether Optimistic Hedge actually converges pointwise*.

Motivated by the afore-described lines of work, and the importance of last iterate convergence for Game Theory and the modern applications of GDA-style methods in Optimization, our goal in this work is to generalize the results of [8, 11] to the general min-max problem (3), or equivalently (1); indeed, we will focus on the latter, but our algorithms are readily applicable to the former as the two problems are equivalent [1]. With the constraint that  $(\mathbf{x}, \mathbf{y})$  should remain in  $\Delta_n \times \Delta_m$ , GDA and OGDA are not applicable. Indeed, the natural GDA-style method for min-max problems in this case is the celebrated Multiplicative-Weights-Update (MWU) method, which is tantamount to FTRL with entropy-regularization. Unsurprisingly, in the same way that GDA suffers in the unconstrained problem (4), MWU exhibits cycling in the constrained problem (1) (a recent work is [2] and was also shown empirically in [19]). So it is natural for us to study instead its optimistic variant, “Optimistic Multiplicative-Weights-Update (OMWU),” (called Optimistic Hedge in [19]) which corresponds to Optimistic FTRL with entropy-regularization, the equations of which are given in Section 2.2. Our main result is the following (restated as Theorem 5 after Section 2.2) and answers an open question asked in [19] as applicable to two player zero sum games:

► **Theorem 1 (Last-Iterate Convergence of OMWU).** *Whenever (1) has a unique optimal solution  $(\mathbf{x}^*, \mathbf{y}^*)$ , OMWU with a small enough learning rate and initialized at the pair of uniform distributions  $(\frac{1}{n}\mathbf{1}, \frac{1}{m}\mathbf{1})$  exhibits last-iterate convergence to the optimal solution. That is, if  $(\mathbf{x}^t, \mathbf{y}^t)$  are the vectors maintained by OMWU at step  $t$ , then  $\lim_{t \rightarrow \infty} (\mathbf{x}^t, \mathbf{y}^t) = (\mathbf{x}^*, \mathbf{y}^*)$ .*

► **Remark.** We note that the assumption about uniqueness of the optimal solution for problem (1) is generic in the following sense: Within the set of all zero-sum games, the set of zero-sum games with non-unique equilibrium has Lebesgue measure zero [2, 6]. This implies that if  $A$ ’s entries are sampled independently from some continuous distribution, then with probability one the min-max problem (1) will have a unique solution.

Our paper provides *two important messages*:

<sup>4</sup> OGDA is tantamount to Optimistic FTRL with  $\ell_2^2$ -regularization, in the same way that GDA is tantamount to FTRL with  $\ell_2^2$ -regularization; see e.g. [17]. OGDA essentially boils down to GDA with negative momentum.

- It strengthens the intuition that optimism helps the trajectories of learning dynamics stabilize (e.g., Optimistic MWU vs MWU or Optimistic GDA vs GDA; as the papers of Syrgkanis et al [19] and Daskalakis et al [8] also do).
- The techniques we use (typically appear in dynamical systems literature) to prove convergence for the last iterate, are fundamentally different from those commonly used to prove convergence of the time average of a learning algorithm.

**Notation.** Vectors in  $\Delta_n, \Delta_m$  are denoted in boldface  $\mathbf{x}, \mathbf{y}$ . Time indices are denoted by superscripts. Thus, a time indexed vector  $\mathbf{x}$  at time  $t$  is denoted as  $\mathbf{x}^t$ . We use the letter  $J$  to denote the Jacobian of a function (with appropriate subscript),  $\mathbf{I}, \mathbf{0}, \mathbf{1}$  to denote the identity, zero matrix and all ones vector respectively with appropriate subscripts to indicate the size. Moreover,  $(A\mathbf{y})_i$  captures  $\sum_j A_{ij}y_j$ . The support of  $\mathbf{x}$  is denoted by  $\text{Supp}(\mathbf{x})$ . Finally we use  $(\mathbf{x}^*, \mathbf{y}^*)$  to denote the optimal solution for the min-max problem (1) and  $[n]$  to denote  $\{1, \dots, n\}$ .

## 2 Preliminaries

### 2.1 Definitions and facts

#### Dynamical Systems

A recurrence relation of the form  $\mathbf{x}^{t+1} = w(\mathbf{x}^t)$  is a discrete time dynamical system, with update rule  $w : \mathcal{S} \rightarrow \mathcal{S}$  where  $\mathcal{S} = \Delta_n \times \Delta_m \times \Delta_n \times \Delta_m$  for our purposes. The point  $\mathbf{z}$  is called a *fixed point* or *equilibrium* of  $w$  if  $w(\mathbf{z}) = \mathbf{z}$ . We will be interested in the following well known fact that will be used in our proofs.

► **Proposition 2** (e.g. [9]). *If the Jacobian of the update rule  $w^5$  at a fixed point  $\mathbf{z}$  has spectral radius less than one, then there exists a neighborhood  $U$  around  $\mathbf{z}$  such that for all  $\mathbf{x} \in U$ , the dynamics converges to  $\mathbf{z}$ , i.e.,  $\lim_{n \rightarrow \infty} w^n(\mathbf{x}) = \mathbf{z}$ . We call  $w$  a contraction mapping in  $U$ .*

### 2.2 OMWU Method

Our main contribution is that the last iterate of OMWU converges to the optimal solution. The OMWU dynamics is defined as follows ( $t \geq 1$ ):

$$\begin{aligned} x_i^{t+1} &= x_i^t \frac{e^{2\eta(A\mathbf{y}^t)_i - \eta(A\mathbf{y}^{t-1})_i}}{\sum_{j=1}^n x_j^t e^{2\eta(A\mathbf{y}^t)_j - \eta(A\mathbf{y}^{t-1})_j}} \text{ for all } i \in [n], \\ y_i^{t+1} &= y_i^t \frac{e^{-2\eta(A^\top \mathbf{x}^t)_i + \eta(A^\top \mathbf{x}^{t-1})_i}}{\sum_{j=1}^m y_j^t e^{-2\eta(A^\top \mathbf{x}^t)_j + \eta(A^\top \mathbf{x}^{t-1})_j}} \text{ for all } i \in [m]. \end{aligned} \quad (5)$$

Points  $(\mathbf{x}^1, \mathbf{y}^1), (\mathbf{x}^0, \mathbf{y}^0)$  are the initial conditions and are given as input. We call  $0 < \eta < 1$  the *stepsize* of the dynamics. It is more convenient to interpret OMWU dynamics as mapping a quadruple to quadruple  $((\mathbf{x}^t, \mathbf{y}^t, \mathbf{x}^{t-1}, \mathbf{y}^{t-1}) \rightarrow (\mathbf{x}^{t+1}, \mathbf{y}^{t+1}, \mathbf{x}^t, \mathbf{y}^t)$ , see Section 3.2 for the construction of the dynamical system).

► **Remark.** Let  $(\mathbf{x}^*, \mathbf{y}^*)$  be the optimal solution. We see that  $(\mathbf{x}^*, \mathbf{y}^*, \mathbf{x}^*, \mathbf{y}^*)$  is a fixed point of the mapping. Furthermore,  $\Delta_n \times \Delta_m \times \Delta_n \times \Delta_m$  is invariant under OMWU dynamics. For  $t \geq 1$ , if  $x_i^t = 0$  then  $x_i$  remains zero for all times greater than  $t$ , and if it is positive, it remains positive (both numerator and denominator are positive)<sup>6</sup>. In words, at all times the

<sup>5</sup> We assume  $w$  is a continuously differential function.

<sup>6</sup> Same holds for vector  $\mathbf{y}$ .

OMWU satisfies the non-negativity constraints and the renormalization factor (denominator) makes both  $\mathbf{x}, \mathbf{y}$ 's coordinates sum up to one. A last observation is that every fixed point of OMWU dynamics (mapping a quadruple to quadruple) has the form  $(\mathbf{x}, \mathbf{y}, \mathbf{x}, \mathbf{y})$  (two same copies). Equation (8) shows how to express OMWU dynamics as a dynamical system.

### 2.3 Linear Variant of OMWU

We provide the linear variant of OMWU dynamics (5) because we use it in some intermediate lemmas (appear in appendix).

$$\begin{aligned} x_i^{t+1} &= x_i^t \frac{1+2\eta(\mathbf{A}\mathbf{y}^t)_i - \eta(\mathbf{A}\mathbf{y}^{t-1})_i}{\sum_{j=1}^n x_j^t (1+2\eta(\mathbf{A}\mathbf{y}^t)_j - \eta(\mathbf{A}\mathbf{y}^{t-1})_j)} \text{ for all } i \in [n], \\ y_i^{t+1} &= y_i^t \frac{1-2\eta(\mathbf{A}^\top \mathbf{x}^t)_i + \eta(\mathbf{A}^\top \mathbf{x}^{t-1})_i}{\sum_{j=1}^m y_j^t (1-2\eta(\mathbf{A}^\top \mathbf{x}^t)_j + \eta(\mathbf{A}^\top \mathbf{x}^{t-1})_j)} \text{ for all } i \in [m]. \end{aligned} \quad (6)$$

This dynamics is derived by considering the first order approximation of the exponential function. Step size  $\eta$  in this case should be chosen sufficiently small so that both numerator and denominator are positive.

### 2.4 More definitions and statement of our result

► **Definition 3** ([12]). Assume  $\alpha > 0$ . We call a point  $(\mathbf{x}, \mathbf{y}) \in \Delta_n \times \Delta_m$   $\alpha$ -close if for each  $i$  we have that  $x_i \leq \alpha$  or  $|\mathbf{x}^\top \mathbf{A}\mathbf{y} - (\mathbf{A}\mathbf{y})_i| \leq \alpha$  and for each  $j$  it holds  $y_j \leq \alpha$  or  $|\mathbf{x}^\top \mathbf{A}\mathbf{y} - (\mathbf{A}^\top \mathbf{x})_j| \leq \alpha$ .

► **Remark.** Think of  $\alpha$ -close points as  $\alpha$ -approximate optimal solutions for min-max problems that are induced by *submatrices* of  $A$  ( $\alpha$ -approximate stationary points). Moreover, if  $(\mathbf{x}, \mathbf{y})$  is 0-close point does not necessarily imply  $(\mathbf{x}, \mathbf{y})$  is the optimal solution of problem (1)!

► **Definition 4** (Approximate solution). Assume  $\epsilon > 0$ . We call a point  $(\mathbf{x}, \mathbf{y}) \in \Delta_n \times \Delta_m$   $\epsilon$ -approximate (or  $\epsilon$ -approximate Nash equilibrium) if for all  $\tilde{\mathbf{x}} \in \Delta_n$  we get that  $\tilde{\mathbf{x}}^\top \mathbf{A}\mathbf{y} \leq \mathbf{x}^\top \mathbf{A}\mathbf{y} + \epsilon$  (max player deviates) and for all  $\tilde{\mathbf{y}} \in \Delta_m$  we get that  $\mathbf{x}^\top \mathbf{A}\tilde{\mathbf{y}} \geq \mathbf{x}^\top \mathbf{A}\mathbf{y} - \epsilon$  (min player deviates).

► **Remark.** Think of  $\epsilon$ -approximate points as approximate optimal solutions to the min-max problem (1). Moreover, if  $(\mathbf{x}, \mathbf{y})$  is 0-approximate then  $(\mathbf{x}, \mathbf{y})$  is the optimal solution of problem (1).

#### Statement of our results

We finish the preliminary section by stating formally the main result.

► **Theorem 5** (OMWU converges). *Let  $A$  be a  $n \times m$  matrix and assume that*

$$\min_{\mathbf{y} \in \Delta_m} \max_{\mathbf{x} \in \Delta_n} \mathbf{x}^\top \mathbf{A}\mathbf{y}$$

*has a unique solution  $(\mathbf{x}^*, \mathbf{y}^*)$ . It holds that for  $\eta$  sufficiently small (depends on  $n, m, A$ ), starting from the uniform distribution, i.e.,  $(\mathbf{x}^1, \mathbf{y}^1) = (\mathbf{x}^0, \mathbf{y}^0) = (\frac{1}{n}\mathbf{1}, \frac{1}{m}\mathbf{1})$ , it holds*

$$\lim_{t \rightarrow \infty} (\mathbf{x}^t, \mathbf{y}^t) = (\mathbf{x}^*, \mathbf{y}^*),$$

*under OMWU dynamics. The step size  $\eta$  is constant, i.e., does not scale with time<sup>7</sup>.*

<sup>7</sup> Our proof also works if the starting points  $(\mathbf{x}^1, \mathbf{y}^1), (\mathbf{x}^0, \mathbf{y}^0)$  are both in the interior of  $\Delta_n \times \Delta_m$  and not necessarily uniform, however the choice of  $\eta$  depends on the initial distributions as well and not only on  $n, m, A$ .

We need to note that it is not clear from our theorem how small  $\eta$  is and its dependence on the size of  $A$ . Nevertheless, our convergence result holds for constant stepsizes as opposed to the classic no-regret learning literature where  $\eta$  scales like  $\frac{1}{\sqrt{T}}$  after  $T$  iterations. Another result we know of this flavor is about MWU algorithm on congestion games [16].

### 3 Last iterate convergence of OMWU

In this section we show our main result (Theorem 5), by breaking the proof into three key theorems. The first theorem says that KL divergence from the  $t$ -th iterate  $(\mathbf{x}^t, \mathbf{y}^t)$  to the optimal solution  $(\mathbf{x}^*, \mathbf{y}^*)$ , i.e., (sum of KL divergences to be exact)

$$\sum_i x_i^* \ln(x_i^*/x_i^t) + \sum_i y_i^* \ln(y_i^*/y_i^t),$$

decreases with time  $t \geq 2$  by at least a factor of  $\eta^3$  per iteration, unless the iterate  $(\mathbf{x}^t, \mathbf{y}^t)$  is  $O(\eta^{1/3})$ -close (see Definition 3). Moreover, provided that the stepsize  $\eta$  is small enough, we can show the structural result that  $(\mathbf{x}^t, \mathbf{y}^t)$  lies in a neighborhood of  $(\mathbf{x}^*, \mathbf{y}^*)$  that becomes smaller and smaller as  $\eta \rightarrow 0$ . Finally, as long as OMWU dynamics has reached a small neighborhood around  $(\mathbf{x}^*, \mathbf{y}^*)$ , we show that the update rule of the dynamical system induced by OMWU is contracting, and the last iterate convergence result follows. Formally we show:

► **Theorem 6** (KL decreasing). *Let  $(\mathbf{x}^*, \mathbf{y}^*)$  be the unique optimal solution of problem (1) and  $\eta$  sufficiently small. Then*

$$D_{KL}((\mathbf{x}^*, \mathbf{y}^*) || (\mathbf{x}^t, \mathbf{y}^t))$$

*is decreasing with time  $t$  by (at least)  $\Omega(\eta^3)$  unless  $(\mathbf{x}^t, \mathbf{y}^t)$  is  $O(\eta^{1/3})$ -close.*

► **Theorem 7** ( $\eta^{1/3}$ -close implies close to optimum in  $\ell_1$ ). *Assume that  $(\mathbf{x}^*, \mathbf{y}^*)$  is unique optimal solution of the problem (1). Let  $T$  (depends on  $\eta$ ) be the first time KL divergence does not decrease by  $\Omega(\eta^3)$ . It follows that as  $\eta \rightarrow 0$ , the  $\eta^{1/3}$ -close point  $(\mathbf{x}^T, \mathbf{y}^T)$  has distance from  $(\mathbf{x}^*, \mathbf{y}^*)$  that goes to zero, i.e.,  $\lim_{\eta \rightarrow 0} \|(\mathbf{x}^*, \mathbf{y}^*) - (\mathbf{x}^T, \mathbf{y}^T)\|_1 = 0$ .*

► **Theorem 8** (OMWU is a contraction). *Let  $(\mathbf{x}^*, \mathbf{y}^*)$  be the unique optimal solution to the min-max problem (1). There exists a neighborhood  $U \subset \Delta_n \times \Delta_m \times \Delta_n \times \Delta_m$  of  $(\mathbf{x}^*, \mathbf{y}^*, \mathbf{x}^*, \mathbf{y}^*)$ <sup>8</sup> so that for all  $(\mathbf{x}^1, \mathbf{y}^1, \mathbf{x}^0, \mathbf{y}^0) \in U$  we have that  $\lim_{t \rightarrow \infty} (\mathbf{x}^t, \mathbf{y}^t, \mathbf{x}^{t-1}, \mathbf{y}^{t-1}) = (\mathbf{x}^*, \mathbf{y}^*, \mathbf{x}^*, \mathbf{y}^*)$  under OMWU dynamics as defined in (5) and (8) (Section 3.2).*

Assuming these three theorems, our main result is straightforward.

**Proof of Theorem 5.** Let  $\eta$  be sufficiently small. If  $(\mathbf{x}^1, \mathbf{y}^1) = (\frac{1}{n}\mathbf{1}, \frac{1}{m}\mathbf{1})$  (starting point is uniform) then an easy upper bound (by removing negative terms) on KL divergence from  $(\mathbf{x}^1, \mathbf{y}^1)$  to  $(\mathbf{x}^*, \mathbf{y}^*)$  is  $-\sum_{i=1}^n x_i^* \log x_i^1 + \sum_{i=1}^m y_i^* \log y_i^1 = \log(nm)$ . Therefore using Theorem 6 we have that after at most  $T$  that is  $O(\frac{\log(nm)}{\eta^3})$  steps, OMWU reaches a  $O(\eta^{1/3})$ -close point ( $T$  is the first time so that KL divergence from current iterate to optimal solution  $(\mathbf{x}^*, \mathbf{y}^*)$  has not decreased by at least a factor of  $\eta^3$ ) or the KL divergence between the optimal solution and  $(\mathbf{x}^T, \mathbf{y}^T)$  is  $O(\eta^3)$  (KL divergence was decreasing by at least a factor of  $\eta^3$  for all iterations until the iterate reached a  $\ell_1$  distance  $O(\eta^3)$ ). In the latter case it

<sup>8</sup> Since  $(\mathbf{x}^*, \mathbf{y}^*, \mathbf{x}^*, \mathbf{y}^*)$  might be on the boundary of  $\Delta_n \times \Delta_m \times \Delta_n \times \Delta_m$ ,  $U$  is the intersection of an open ball around  $(\mathbf{x}^*, \mathbf{y}^*, \mathbf{x}^*, \mathbf{y}^*)$  with  $\Delta_n \times \Delta_m \times \Delta_n \times \Delta_m$ .

follows  $\|(\mathbf{x}^*, \mathbf{y}^*) - (\mathbf{x}^T, \mathbf{y}^T)\|_1^2$  is  $O(\eta^3)$  and hence  $(\mathbf{x}^T, \mathbf{y}^T)$  is  $O(\eta^{3/2})$  in  $\ell_1$  distance from the optimal solution, therefore for small  $\eta$ ,  $(\mathbf{x}^{T+1}, \mathbf{y}^{T+1}, \mathbf{x}^T, \mathbf{y}^T)$  is in the neighborhood  $U$  that is needed for contraction (Theorem 8). In the former case, by Theorem 7 (for  $\eta$  sufficiently small) it follows that  $(\mathbf{x}^{T+1}, \mathbf{y}^{T+1}, \mathbf{x}^T, \mathbf{y}^T)$  is also in the neighborhood  $U$  that is needed for contraction (Theorem 8)<sup>9</sup>. The proof follows by Theorem 8.  $\blacktriangleleft$

In the next subsections we will provide the proofs to all three key theorems.

### 3.1 KL decreases and OMWU reaches neighborhood

In this subsection we argue about the proofs of Theorems 6 and 7. The inequality we managed to prove (see in the appendix the proof of Theorem 6) is the following:

$$\begin{aligned} & D_{KL}((\mathbf{x}^*, \mathbf{y}^*) \| (\mathbf{x}^{t+1}, \mathbf{y}^{t+1})) - D_{KL}((\mathbf{x}^*, \mathbf{y}^*) \| (\mathbf{x}^t, \mathbf{y}^t)) \leq \\ & - \sum_{i=1}^n x_i^t \left( \left( \frac{1}{2} - O(\eta) \right) \eta^2 \left( 2(A\mathbf{y}^t)_i - 2\mathbf{x}^{t \top} A\mathbf{y}^t - (A\mathbf{y}^{t-1})_i + \mathbf{x}^{t \top} A\mathbf{y}^{t-1} \right)^2 \right) \\ & - \sum_{i=1}^m y_i^t \left( \left( \frac{1}{2} - O(\eta) \right) \eta^2 \left( 2(A^\top \mathbf{x}^t)_i - 2\mathbf{x}^{t \top} A\mathbf{y}^t - (A^\top \mathbf{x}^{t-1})_i + \mathbf{x}^{t-1 \top} A\mathbf{y}^t \right)^2 \right) + O(\eta^3). \end{aligned} \quad (7)$$

The proof of the inequality is quite long, we choose to provide intuition and skip the details. We refer to the appendix for a proof. The inequality says that OMWU dynamics has a good progress (KL divergence decreases by at least a factor of  $\eta^3$ ) as long as the current and previous iterate  $(\mathbf{x}^t, \mathbf{y}^t)$ ,  $(\mathbf{x}^{t-1}, \mathbf{y}^{t-1})$  are not  $\alpha$ -close for  $\alpha$  chosen to be  $O(\eta^{1/3})$ . This situation appears a lot in gradient methods when the dynamics is close to a stationary point, the gradient of  $f$  is small and the progress is small as opposed to the case where the gradient of  $f$  is big and there is satisfying progress. The RHS of inequality (7) captures the “distance” from stationarity. Thus, as long as we are not close to a stationary point (i.e.,  $O(\eta^{1/3})$ -close) in a time window between  $1, 2, \dots, k$ , KL divergence from current iterate ( $k$ -th) to the optimum has decreased by (at least)  $\Omega(k\eta^3)$  compared to KL divergence from first iterate to the optimum.

Moreover, suppose that at some point of OMWU dynamics, KL divergence from current iterate to the optimum did not decrease by at least a factor of  $\eta^3$  and let  $T$  be the iteration this happened. As we have already argued,  $(\mathbf{x}^T, \mathbf{y}^T)$  is a  $O(\eta^{1/3})$ -close point. We can show that as long as  $\eta$  is sufficiently small, then for all  $i, j$  in the support of  $(\mathbf{x}^*, \mathbf{y}^*)$ ,  $x_i^T, y_j^T$  are (at least)  $\Omega(\eta^{1/3})$  i.e., coordinates in the support of the optimum will have non negligible probability in  $(\mathbf{x}^T, \mathbf{y}^T)$ . Formally:

► **Lemma 9.** *Let  $i \in \text{Supp}(\mathbf{x}^*)$  and  $j \in \text{Supp}(\mathbf{y}^*)$ . It holds that  $x_i^T \geq \frac{1}{2}\eta^{1/3}$  and  $y_j^T \geq \frac{1}{2}\eta^{1/3}$  as long as*

$$\eta^{1/3} \leq \min_{s \in \text{Supp}(\mathbf{x}^*)} \frac{1}{(nm)^{1/x_s^*}}, \min_{s \in \text{Supp}(\mathbf{y}^*)} \frac{1}{(nm)^{1/y_s^*}}.$$

**Proof.** By definition of  $T$ , the KL divergence is decreasing for  $2 \leq t \leq T-1$ , thus

$$D_{KL}((\mathbf{x}^*, \mathbf{y}^*) \| (\mathbf{x}^{T-1}, \mathbf{y}^{T-1})) < D_{KL}((\mathbf{x}^*, \mathbf{y}^*) \| (\mathbf{x}^1, \mathbf{y}^1)).$$

Therefore  $x_i^* \log \frac{1}{x_i^{T-1}} < \sum_i x_i^* \log \frac{1}{x_i^1} + \sum_i y_j^* \log \frac{1}{y_j^1} = \log(mn)$ . It follows that  $x_i^T > 1/(mn)^{\frac{1}{x_i^*}} \geq \eta^{1/3}$  for  $x_i^* > 0$  ( $i \in \text{Supp}(\mathbf{x}^*)$ ). Since  $|x_i^T - x_i^{T-1}|$  is  $O(\eta)$  (Lemma 11) the result follows. Similarly, the argument works for  $y_j^T$ .  $\blacktriangleleft$

<sup>9</sup> In both cases we used that iterate  $(\mathbf{x}^T, \mathbf{y}^T)$  and  $(\mathbf{x}^{T+1}, \mathbf{y}^{T+1})$  have  $\ell_1$  distance  $O(\eta)$ , this is Lemma 11.



Lemma 9 indicates that the stepsize  $\eta$  might have to be exponentially small in the dimension (OMWU dynamics is slow when  $\eta$  is very small). We can now prove Theorem 7.

**Proof of Theorem 7.** From Lemma 9 and definition of  $T$ , we get that  $|(A\mathbf{y}^T)_i - \mathbf{x}^T \top A\mathbf{y}^T|$  is  $O(\eta^{1/3})$  for all  $i$  in the support of  $\mathbf{x}^*$  and  $|(A^\top \mathbf{x}^T)_j - \mathbf{x}^T \top A\mathbf{y}^T|$  is  $O(\eta^{1/3})$  for all  $j$  in the support of  $\mathbf{y}^*$ . We consider  $(\mathbf{w}^T, \mathbf{z}^T)$  to be the projection of the point  $(\mathbf{x}^T, \mathbf{y}^T)$  by removing all the coordinates that have probability mass less than  $\frac{1}{2}\eta^{1/3}$  and rescale so that the coordinates sum up to one.

We restrict ourselves to the corresponding subproblem (submatrix). It is clear that  $(\mathbf{w}^T, \mathbf{z}^T)$  is a  $O(\eta^{1/3})$ -approximate solution<sup>10</sup> for the subproblem. Let  $v = \mathbf{x}^* A \mathbf{y}^*$  be the optimal value. By uniqueness of the optimal solution, we get that  $(A\mathbf{y}^*)_i = v$  for all  $i \in \text{Supp}(\mathbf{x}^*)$  and  $(A\mathbf{y}^*)_i < v$  otherwise (check Lemma C.3 in paper [13] for a proof, where they use Farkas' lemma to show it, we use this fact later in Section 3.2). Similarly  $(A^\top \mathbf{x}^*)_j = v$  for the min player  $\mathbf{y}$  if  $j$  lies in the support of  $\mathbf{y}^*$  and  $(A^\top \mathbf{x}^*)_j > v$  otherwise. We choose  $\eta$  so small that every  $O(\eta^{1/3})$ -approximate solution  $(\mathbf{x}, \mathbf{y})$  has the property that  $(A\mathbf{y})_i \leq v - \eta^{1/4}$ ,  $(A^\top \mathbf{x})_j \geq v + \eta^{1/4}$  for all  $i \notin \text{Supp}(\mathbf{x}^*)$  and  $j \notin \text{Supp}(\mathbf{y}^*)$  respectively (this is possible by continuity of the bilinear function and Claim 10 below). Hence we conclude that if  $\eta$  is small enough, the coordinates in the vector  $(\mathbf{w}^T, \mathbf{z}^T)$  that are not in the support of the optimal solution (since  $\eta^{1/4} \gg \eta^{1/3}$ ), should have probability mass  $O(\eta^{1/3})$  at time  $T$ .

► **Claim 10.** *Let  $(\mathbf{x}^*, \mathbf{y}^*)$  be the unique optimal solution to the problem (1). For every  $\epsilon > 0$ , there exists an  $\delta(\epsilon) > 0$  so that for every  $\delta$ -approximate solution  $(\mathbf{x}, \mathbf{y})$  we get that  $|x_i - x_i^*| < \epsilon$  for all  $i \in [n]$ . Analogously holds for player  $\mathbf{y}$ .*

**Proof.** We will prove this by contradiction. Assume there is an  $\epsilon$  that violates this statement. We choose a sequence  $\delta_k$  so that  $\lim_{k \rightarrow \infty} \delta_k = 0$  and also there is a sequence  $(\mathbf{x}_k, \mathbf{y}_k)$  of  $\delta_k$ -approximate Nash equilibrium with  $|x_{k,i} - x_i^*| \geq \epsilon$  for some strategy  $i$ . Since  $\Delta_n \times \Delta_m$  is compact and the sequence above is bounded, there is a convergent subsequence. The limit of the convergent subsequence is a Nash equilibrium by definition of  $\delta$ -approximate (Definition 4). By uniqueness it follows that the  $i$ -th coordinate of the convergent sequence must converge to  $x_i^*$ , hence we reached a contradiction. ◀

Therefore, if we restrict to the subproblem induced by the strategies in the support of  $(\mathbf{x}^*, \mathbf{y}^*)$ , the projected vector  $(\mathbf{w}^T, \mathbf{z}^T)$  is a  $O(\eta^{1/3})$ -approximate solution of the subgame.

From Claim 10, as  $\eta \rightarrow 0$  it follows that the  $\ell_1$  distance (any distance suffices) between the projected  $(\mathbf{w}^T, \mathbf{z}^T)$  and the optimal solution (Nash equilibrium) of the subgame goes to zero. Since the optimal solution of the subgame is exactly the same as the optimal solution of the original game we get that as  $\eta \rightarrow 0$ ,  $(\mathbf{x}^T, \mathbf{y}^T)$  reaches  $(\mathbf{x}^*, \mathbf{y}^*)$ . In particular, since  $\|(\mathbf{x}^{T+1}, \mathbf{y}^{T+1}) - (\mathbf{x}^T, \mathbf{y}^T)\|_1$  is  $O(\eta)$  (see Lemma 11) there exists a  $\eta$  small so that  $(\mathbf{x}^{T+1}, \mathbf{y}^{T+1}, \mathbf{x}^T, \mathbf{y}^T)$  is inside the necessary neighborhood  $U$  of  $(\mathbf{x}^*, \mathbf{y}^*, \mathbf{x}^*, \mathbf{y}^*)$  that gives contraction (Theorem 8). ◀

### 3.2 Proving contraction

The purpose of this section is to prove Theorem 8. To show contraction of OMWU dynamics in a neighborhood of the optimal solution  $(\mathbf{x}^*, \mathbf{y}^*)$ , we first construct a dynamical system that captures OMWU. Moreover, we prove that the Jacobian of the update rule of that particular

<sup>10</sup>By  $\epsilon$ -approximate optimal solution we mean the  $\epsilon$ -approximate Nash equilibrium notion (additive), see Definition 4.

dynamical system computed at the optimal solution, has spectral radius less than one. This suffices to prove contraction (see Proposition 2). As a result, as long as OMWU reaches a small neighborhood of  $(\mathbf{x}^*, \mathbf{y}^*, \mathbf{x}^*, \mathbf{y}^*)$ , it converges pointwise (last iterate convergence) to it<sup>11</sup>. Below we provide the update rule  $g$  of the dynamical system, which consists of 4 components:

$$\begin{aligned} g(\mathbf{x}, \mathbf{y}, \mathbf{z}, \mathbf{w}) &:= (g_1(\mathbf{x}, \mathbf{y}, \mathbf{z}, \mathbf{w}), g_2(\mathbf{x}, \mathbf{y}, \mathbf{z}, \mathbf{w}), g_3(\mathbf{x}, \mathbf{y}, \mathbf{z}, \mathbf{w}), g_4(\mathbf{x}, \mathbf{y}, \mathbf{z}, \mathbf{w})), \\ g_{1,i}(\mathbf{x}, \mathbf{y}, \mathbf{z}, \mathbf{w}) &:= (g_1(\mathbf{x}, \mathbf{y}, \mathbf{z}, \mathbf{w}))_i := x_i \frac{e^{2\eta(A\mathbf{y})_i - \eta(A\mathbf{w})_i}}{\sum_t x_t e^{2\eta(A\mathbf{y})_t - \eta(A\mathbf{w})_t}} \text{ for all } i \in [n], \\ g_{2,i}(\mathbf{x}, \mathbf{y}, \mathbf{z}, \mathbf{w}) &:= (g_2(\mathbf{x}, \mathbf{y}, \mathbf{z}, \mathbf{w}))_i := y_i \frac{e^{-2\eta(A^\top \mathbf{x})_i + \eta(A^\top \mathbf{z})_i}}{\sum_t y_t e^{-2\eta(A^\top \mathbf{x})_t + \eta(A^\top \mathbf{z})_t}} \text{ for all } i \in [m], \\ g_3(\mathbf{x}, \mathbf{y}, \mathbf{z}, \mathbf{w}) &:= \mathbf{I}_{n \times n} \mathbf{x}, \\ g_4(\mathbf{x}, \mathbf{y}, \mathbf{z}, \mathbf{w}) &:= \mathbf{I}_{m \times m} \mathbf{y}. \end{aligned} \quad (8)$$

It is not hard to check that

$$(\mathbf{x}_{t+1}, \mathbf{y}_{t+1}, \mathbf{x}_t, \mathbf{y}_t) = g(\mathbf{x}_t, \mathbf{y}_t, \mathbf{x}_{t-1}, \mathbf{y}_{t-1}),$$

so  $g$  captures exactly the dynamics of OMWU (5). The equations of the Jacobian of  $g$  can be found in the appendix (see Section A).

### Spectral analysis the Jacobian of OMWU at the optimal solution

The rest of the section constitutes the proof of Theorem 8. Assume  $v = \mathbf{x}^{*\top} A \mathbf{y}^*$ , i.e.,  $v$  is the value of the bilinear function  $\mathbf{x}^\top A \mathbf{y}$  at the optimal solution. We will analyze the Jacobian computed at  $(\mathbf{x}^*, \mathbf{y}^*, \mathbf{x}^*, \mathbf{y}^*)$ <sup>12</sup>.

Assume  $i \notin \text{Supp}(\mathbf{x}^*)$ , then

$$\frac{\partial g_{1,i}}{\partial x_i} = \frac{e^{\eta(A\mathbf{y}^*)_i}}{\sum_t x_t^* e^{\eta(A\mathbf{y}^*)_t}} = \frac{e^{\eta(A\mathbf{y}^*)_i}}{e^{\eta v}}$$

and all other partial derivatives of  $g_{1,i}$  are zero, thus  $\frac{e^{\eta(A\mathbf{y}^*)_i}}{e^{\eta v}}$  is an eigenvalue of the Jacobian computed at  $(\mathbf{x}^*, \mathbf{y}^*, \mathbf{x}^*, \mathbf{y}^*)$ . Moreover because of uniqueness of the optimal solution, it holds that  $\frac{e^{\eta(A\mathbf{y}^*)_i}}{e^{\eta v}} < 1$  because  $(A\mathbf{y}^*)_i - v < 0$  (check Lemma C.3 in [13] for a proof, where they use Farkas' Lemma to show it). Similarly, it holds for  $j \notin \text{Supp}(\mathbf{y}^*)$  that  $\frac{\partial g_{2,j}}{\partial y_j} = \frac{e^{-\eta(A^\top \mathbf{x}^*)_j}}{e^{-\eta v}} < 1$  (again by C.3 in [13] it holds that  $(A\mathbf{x}^*)_j - v > 0$ ) and all other partial derivatives of  $g_{2,j}$  are zero, hence  $\frac{e^{-\eta(A^\top \mathbf{x}^*)_j}}{e^{-\eta v}}$  is an eigenvalue of the Jacobian computed at the optimal solution.

Let  $D_x$  be the diagonal matrix of size  $|\text{Supp}(\mathbf{x}^*)| \times |\text{Supp}(\mathbf{x}^*)|$  that has on the diagonal the nonzero entries of  $\mathbf{x}^*$  and similarly we define  $D_y$  of size  $|\text{Supp}(\mathbf{y}^*)| \times |\text{Supp}(\mathbf{y}^*)|$ . We set  $k_1 = |\text{Supp}(\mathbf{x}^*)|$ ,  $k_2 = |\text{Supp}(\mathbf{y}^*)|$  and  $k = k_1 + k_2$ . Let  $\mathbf{x}', \mathbf{y}'$  be the optimal solution to the min-max problem with payoff matrix the corresponding submatrix of payoff matrix  $A$  (denoted by  $B$ ) after removing the rows/columns which correspond to the coordinates that are not in the support of the unique optimal solution  $(\mathbf{x}^*, \mathbf{y}^*)$ <sup>13</sup>. We consider the submatrix  $\tilde{J}$  of the Jacobian matrix that is created by removing rows and columns of the corresponding coordinates that are not in the support of optimum (for the variables  $\mathbf{x}$  and  $\mathbf{y}$ , these are exactly  $n + m - k$ ). It is clear from above, that the Jacobian of OMWU has eigenvalues

<sup>11</sup> Since the dynamical system is from a quadruple to a quadruple, it is a neighborhood of  $(\mathbf{x}^*, \mathbf{y}^*, \mathbf{x}^*, \mathbf{y}^*)$ .

<sup>12</sup> See also Equations (14) of the Jacobian computed at  $(\mathbf{x}^*, \mathbf{y}^*, \mathbf{x}^*, \mathbf{y}^*)$ .

<sup>13</sup> Note that  $(\mathbf{x}', \mathbf{y}')$  should be the *unique* optimal solution to the min-max problem with payoff matrix  $B$ .

with absolute value less than one iff  $\tilde{J}$  has as well. After also removing the rows (and the corresponding columns) that have only zero entries (these are exactly  $n + m - k$ , result zero eigenvalues and correspond to variables  $\mathbf{z}$  and  $\mathbf{w}$ ) the resulting submatrix (denote it by  $J$ ) boils down to the following matrix  $J$ :

$$\begin{pmatrix} \mathbf{I}_{k_1 \times k_1} - D_x \mathbf{1}_{k_1} \mathbf{1}_{k_1}^\top & 2\eta D_x (B - v \mathbf{1}_{k_1} \mathbf{1}_{k_2}^\top) & \mathbf{0}_{k_1 \times k_1} & -\eta D_x (B - v \mathbf{1}_{k_1} \mathbf{1}_{k_2}^\top) \\ 2\eta D_y (v \mathbf{1}_{k_2} \mathbf{1}_{k_1}^\top - B^\top) & \mathbf{I}_{k_2 \times k_2} - D_y \mathbf{1}_{k_2} \mathbf{1}_{k_2}^\top & -\eta D_y (v \mathbf{1}_{k_2} \mathbf{1}_{k_1}^\top - B^\top) & \mathbf{0}_{k_2 \times k_2} \\ \mathbf{I}_{k_1 \times k_1} & \mathbf{0}_{k_1 \times k_2} & \mathbf{0}_{k_1 \times k_1} & \mathbf{0}_{k_1 \times k_2} \\ \mathbf{0}_{k_2 \times k_1} & \mathbf{I}_{k_2 \times k_2} & \mathbf{0}_{k_2 \times k_1} & \mathbf{0}_{k_2 \times k_2} \end{pmatrix}. \quad (9)$$

It is clear that  $(\mathbf{1}_{k_1}, \mathbf{0}_{k_2}, \mathbf{0}_{k_1}, \mathbf{0}_{k_2})$ ,  $(\mathbf{0}_{k_1}, \mathbf{1}_{k_2}, \mathbf{0}_{k_1}, \mathbf{0}_{k_2})$  are left eigenvectors with eigenvalues zero and thus any right eigenvector  $(\tilde{\mathbf{x}}, \tilde{\mathbf{y}}, \tilde{\mathbf{z}}, \tilde{\mathbf{w}})$  with nonzero eigenvalue has the property that  $\tilde{\mathbf{x}}^\top \mathbf{1}_{k_1} = 0$  and  $\tilde{\mathbf{y}}^\top \mathbf{1}_{k_2} = 0$ . Hence every nonzero eigenvalue of the matrix above is an eigenvalue of the matrix below:

$$J_{\text{new}} = \begin{pmatrix} \mathbf{I}_{k_1 \times k_1} & 2\eta D_x B & \mathbf{0}_{k_1 \times k_1} & -\eta D_x B \\ -2\eta D_y B^\top & \mathbf{I}_{k_2 \times k_2} & \eta D_y B^\top & \mathbf{0}_{k_2 \times k_2} \\ \mathbf{I}_{k_1 \times k_1} & \mathbf{0}_{k_1 \times k_2} & \mathbf{0}_{k_1 \times k_1} & \mathbf{0}_{k_1 \times k_2} \\ \mathbf{0}_{k_2 \times k_1} & \mathbf{I}_{k_2 \times k_2} & \mathbf{0}_{k_2 \times k_1} & \mathbf{0}_{k_2 \times k_2} \end{pmatrix}. \quad (10)$$

Let  $p(\lambda)$  be the characteristic polynomial of the matrix (10). After row/column operations it boils down to

$$(-1)^k \det \begin{pmatrix} \lambda(1-\lambda)\mathbf{I}_{k_1 \times k_1} & (2\lambda-1)\eta D_x B \\ -\eta(2\lambda-1)D_y B^\top & \lambda(1-\lambda)\mathbf{I}_{k_2 \times k_2} \end{pmatrix} = (1-2\lambda)^k q \left( \frac{\lambda(\lambda-1)}{2\lambda-1} \right), \quad (11)$$

where  $q(\lambda)$  is the characteristic polynomial of

$$J_{\text{small}} = \begin{pmatrix} \mathbf{0}_{k_1 \times k_1} & \eta D_x B \\ -\eta D_y B^\top & \mathbf{0}_{k_2 \times k_2} \end{pmatrix}. \quad (12)$$

Observe that

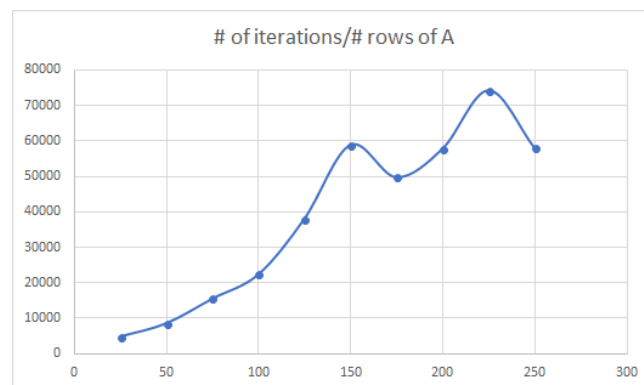
$$J_{\text{small}} \cdot \begin{pmatrix} D_x & \mathbf{0}_{k_1 \times k_2} \\ \mathbf{0}_{k_2 \times k_1} & D_y \end{pmatrix} \text{ is real skew symmetric,}$$

and hence by Lemma 16,  $J_{\text{small}}$  has eigenvalues of the form<sup>14</sup>  $\pm i\eta\tau$  with  $\tau \in \mathbb{R}$  (i.e., imaginary eigenvalues; we include  $\eta$  in the expression to conclude that  $\sigma := \eta\tau$  can be sufficiently small in absolute value). We conclude that any nonzero eigenvalue  $\lambda$  of the matrix  $J$  should satisfy the equation  $\frac{\lambda(\lambda-1)}{2\lambda-1} = i\sigma$  for some small in absolute value  $\sigma \in \mathbb{R}$ . Finally we get that

$$\lambda = \frac{1 + 2i\sigma \pm \sqrt{1 - 4\sigma^2}}{2}.$$

We compute the square of the magnitude of  $\lambda$  and we get  $|\lambda|^2 = \frac{2-4\sigma^2 \pm 2\sqrt{1-4\sigma^2} + 4\sigma^2}{4} = \frac{1 \pm \sqrt{1-4\sigma^2}}{2} < 1$  unless  $\sigma = 0$  (i.e.,  $\tau = 0$ ). If  $\sigma = 0$ , it means that  $J_{\text{new}}$  has an eigenvalue which is equal to one. Assume that  $(\tilde{\mathbf{x}}, \tilde{\mathbf{y}}, \tilde{\mathbf{z}}, \tilde{\mathbf{w}})$  is the corresponding right eigenvector, it holds that  $B\tilde{\mathbf{y}} = \mathbf{0}$  and  $B^\top \tilde{\mathbf{x}} = \mathbf{0}$ . Assume also that there exists an eigenvalue that is equal to one in the original matrix  $J$ . It follows that  $\mathbf{1}_{k_2}^\top \tilde{\mathbf{y}} = 0$  and  $\mathbf{1}_{k_1}^\top \tilde{\mathbf{x}} = 0$ . It holds that  $\tilde{\mathbf{x}} = \mathbf{0}_{k_1}$  and  $\tilde{\mathbf{y}} = \mathbf{0}_{k_2}$  otherwise  $(\mathbf{x}', \mathbf{y}') + t(\tilde{\mathbf{x}}, \tilde{\mathbf{y}})$  would be another optimal solution (for the min-max problem with payoff matrix  $B$ ; by padding zeros to the vector, we could create another optimal solution for the original min-max problem with payoff matrix  $A$ ) for small enough  $t$ . We reached contradiction because we have assumed uniqueness. Hence all the eigenvalues of  $J$  are less than 1, i.e., the mapping is a contraction mapping and the proof is complete.

<sup>14</sup>We denote  $i = \sqrt{-1}$ .



**Figure 1** In the  $x$  axis we have the number of rows of a square matrix  $A$  and on  $y$  axis the number of iterations of OMWU. This figure captures how the number of iterations depends on the dimensionality of the min-max problem.

## 4 Experiments

The purpose of our experiments is primarily to understand how the speed of convergence of OMWU dynamics (5) scales with the size of matrix  $A$ . Moreover, for  $A$  of fixed size, we are interested in how the speed of convergence scales with the error of the output of OMWU dynamics. By error we mean the  $\ell_1$  distance between the last iterate of OMWU and the optimal solution.

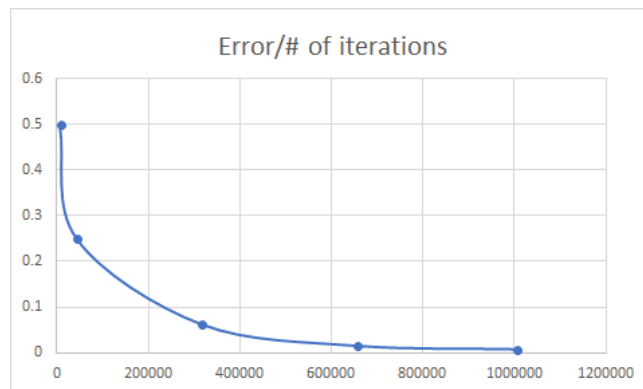
For the former case, we fix the error to be 0.1 and we run OMWU for  $n = 25, 50, \dots, 250$  where the input matrix  $A$  has size  $n \times n$  with entries i.i.d random variables sampled from uniform  $[-1, 1]$ . We output the number of iterations OMWU needs starting from uniform  $(\frac{1}{n}, \dots, \frac{1}{n})$  to reach a solution that is at most 0.1 away from optimal in  $\ell_1$  distance. We note that we computed the optimal solutions using LP-solvers.

For the latter case, we fix  $n = 50$  and we consider the error  $\epsilon$  to be  $\{0.5, 0.25, 0.0625, 0.015625, 0.007812\}$ . Starting from uniform distribution, we count the number of iterations to reach error  $\epsilon$ . The stepsize  $\eta$  is fixed at 0.01 at all times. The results can be found in Figures 1 and 2. If we had to guess, it seems that the relation between dimension and iterations is between linear and quadratic (i.e., OMWU dynamics has roughly cubic-quartic running time in  $n$  if we count the cost of each iteration as quadratic) and the dependence between error  $\epsilon$  and iterations  $t$  seems like  $t$  is inverse polynomial in  $\epsilon$ .

We note the importance of stepsize  $\eta$ .  $\eta$  must be sufficiently small for our proofs to work. If  $\eta$  is chosen to be big, then OMWU might not converge (might cycle, we observed such behavior in experiments). On the other hand, the smaller  $\eta$  is chosen, the smaller the progress of OMWU dynamics (see the inequality claim for KL divergence) and hence the slower the dynamics.

## 5 Conclusion

In this paper we showed that a no-regret algorithm called Optimistic Multiplicative Weights Update (OMWU) converges pointwise to a Nash equilibrium in two player zero sum games (See also a concurrent work to ours [14], in which the authors provide a pointwise result about other dynamics, using different techniques). Our analysis is novel and does not follow the standard approaches of the literature of no-regret learning. We believe that our techniques can be useful in the analysis of other learning algorithms with no provable guarantees of pointwise convergence.



■ **Figure 2** In the  $x$  axis we have the number of iterations of OMWU and on  $y$  axis the  $\ell_1$  distance from the optimal solution. This figure captures how the number of iterations scales with the error.

One interesting open question is to show that OMWU algorithm converges in polynomial time in  $n, m$  (for proper choice of stepsize  $\eta$ ) and find exact rates of convergence. Another possible future direction is to generalize our results about OMWU beyond the bilinear setting.

---

## References

- 1 Ilan Adler. The equivalence of linear programs and zero-sum games. In *International Journal of Game Theory*, pages 165–177, 2013.
- 2 James P. Bailey and Georgios Piliouras. Multiplicative Weights Update in Zero-Sum Games. In *Proceedings of the 2018 ACM Conference on Economics and Computation, Ithaca, NY, USA, June 18-22, 2018*, pages 321–338, 2018.
- 3 David Blackwell. An analog of the minimax theorem for vector payoffs. In *Pacific J. Math.*, pages 1–8, 1956.
- 4 G.W Brown. Iterative Solutions of Games by Fictitious Play. In *Activity Analysis of Production and Allocation*, 1951.
- 5 Nikolo Cesa-Bianchi and Gabor Lugosi. *Prediction, Learning, and Games*. Cambridge University Press, 2006.
- 6 Eric Van Damme. *Stability and perfection of Nash equilibria*. Springer, 1991.
- 7 George B Dantzig. A proof of the equivalence of the programming problem and the game problem. *Activity analysis of production and allocation*, pages 330–338, 1951.
- 8 Constantinos Daskalakis, Andrew Ilyas, Vasilis Syrgkanis, and Haoyang Zeng. Training GANs with Optimism. In *Proceedings of ICLR*, 2018. [arXiv:1711.00141](https://arxiv.org/abs/1711.00141).
- 9 Oded Galor. *Discrete Dynamical Systems*. Springer, 2007.
- 10 Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014.
- 11 Tengyuan Liang and James Stokes. Interaction Matters: A Note on Non-asymptotic Local Convergence of Generative Adversarial Networks. *arXiv preprint:1802.06132*, 2018.
- 12 Ruta Mehta, Ioannis Panageas, Georgios Piliouras, Prasad Tetali, and Vijay V. Vazirani. Mutation, Sexual Reproduction and Survival in Dynamic Environments. In *8th Innovations in Theoretical Computer Science Conference, ITCSC 2017, January 9-11, 2017, Berkeley, CA, USA*, pages 16:1–16:29, 2017.
- 13 Panayotis Mertikopoulos, Christos Papadimitriou, and Georgios Piliouras. Cycles in Adversarial Regularized Learning. In *Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2018, New Orleans, LA, USA, January 7-10, 2018*, pages 2703–2717, 2018.

- 14 Panayotis Mertikopoulos, Houssam Zenati, Bruno Lecuat, Chuan-Sheng Foo, Vijay Chandrasekhar, and Georgios Piliouras. Mirror descent in saddle-point problems: Going the extra (gradient) mile. *CoRR*, abs/1807.02629, 2018.
- 15 Yurii Nesterov. Smooth minimization of non-smooth functions. *Math. Program.*, 103(1):127–152, 2005.
- 16 Gerasimos Palaiopanos, Ioannis Panageas, and Georgios Piliouras. Multiplicative Weights Update with Constant Step-Size in Congestion Games: Convergence, Limit Cycles and Chaos. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA*, pages 5874–5884, 2017.
- 17 Alexander Rakhlin and Karthik Sridharan. Online Learning with Predictable Sequences. In *COLT 2013 - The 26th Annual Conference on Learning Theory, June 12-14, 2013, Princeton University, NJ, USA*, pages 993–1019, 2013.
- 18 J. Robinson. An Iterative Method of Solving a Game. In *Annals of Mathematics*, pages 296–301, 1951.
- 19 Vasilis Syrgkanis, Alekh Agarwal, Haipeng Luo, and Robert E. Schapire. Fast Convergence of Regularized Learning in Games. In *Annual Conference on Neural Information Processing Systems 2015*, pages 2989–2997, 2015.

## A Equations of the Jacobian of OMWU dynamics

### A.1 Equations computed at point $(\mathbf{x}, \mathbf{y}, \mathbf{z}, \mathbf{w})$

Set  $S_x = \sum_{t=1}^n x_t e^{2\eta(A\mathbf{y})_t - \eta(A\mathbf{w})_t}$ ,  $S_y = \sum_{t=1}^m y_t e^{-2\eta(A^\top \mathbf{x})_t + \eta(A^\top \mathbf{z})_t}$  and let  $i, j$  be arbitrary indexes ( $g_{1,i}$  captures the  $i$ -th coordinate of function  $g_1$  etc),

$$\begin{aligned}
\frac{\partial g_{1,i}}{\partial x_i} &= \frac{e^{2\eta(A\mathbf{y})_i - \eta(A\mathbf{w})_i}}{S_x} - x_i \frac{(e^{2\eta(A\mathbf{y})_i - \eta(A\mathbf{w})_i})^2}{S_x^2} \text{ for all } i \in [n], \\
\frac{\partial g_{1,i}}{\partial x_j} &= -x_i e^{2\eta(A\mathbf{y})_j - \eta(A\mathbf{w})_j} \cdot \frac{e^{2\eta(A\mathbf{y})_i - \eta(A\mathbf{w})_i}}{S_x^2} \text{ for all } i \in [n], j \in [n] \text{ and } j \neq i, \\
\frac{\partial g_{1,i}}{\partial y_j} &= x_i e^{2\eta(A\mathbf{y})_i - \eta(A\mathbf{w})_i} \cdot \frac{2\eta A_{ij} S_x - 2\eta \sum_t A_{tj} x_t e^{2\eta(A\mathbf{y})_t - \eta(A\mathbf{w})_t}}{S_x^2} \text{ for all } i \in [n], j \in [m], \\
\frac{\partial g_{1,i}}{\partial z_j} &= 0 \text{ for all } i, j \in [n], \\
\frac{\partial g_{1,i}}{\partial w_j} &= x_i e^{2\eta(A\mathbf{y})_i - \eta(A\mathbf{w})_i} \cdot \frac{-\eta A_{ij} S_x + \eta \sum_t A_{tj} x_t e^{2\eta(A\mathbf{y})_t - \eta(A\mathbf{w})_t}}{S_x^2} \text{ for all } i \in [n], j \in [m], \\
\frac{\partial g_{2,i}}{\partial y_i} &= \frac{e^{-2\eta(A^\top \mathbf{x})_i + \eta(A^\top \mathbf{z})_i}}{S_y} - y_i \frac{(e^{-2\eta(A^\top \mathbf{x})_i + \eta(A^\top \mathbf{z})_i})^2}{S_y^2} \text{ for all } i \in [m], \\
\frac{\partial g_{2,i}}{\partial y_j} &= -y_i e^{-2\eta(A^\top \mathbf{x})_j + \eta(A^\top \mathbf{z})_j} \cdot \frac{e^{-2\eta(A^\top \mathbf{x})_i + \eta(A^\top \mathbf{z})_i}}{S_y^2} \text{ for all } i \in [m], j \in [m] \text{ and } j \neq i, \\
\frac{\partial g_{2,i}}{\partial x_j} &= y_i e^{-2\eta(A^\top \mathbf{x})_i + \eta(A^\top \mathbf{z})_i} \cdot \frac{-2\eta A_{ij}^\top S_y + 2\eta \sum_t A_{tj}^\top y_t e^{-2\eta(A^\top \mathbf{x})_t + \eta(A^\top \mathbf{z})_t}}{S_y^2} \text{ for all } i \in [m], j \in [n], \\
\frac{\partial g_{2,i}}{\partial z_j} &= y_i e^{-2\eta(A^\top \mathbf{x})_i + \eta(A^\top \mathbf{z})_i} \cdot \frac{\eta A_{ij}^\top S_y - \eta \sum_t A_{tj}^\top x_t e^{-2\eta(A^\top \mathbf{x})_t + \eta(A^\top \mathbf{z})_t}}{S_y^2} \text{ for all } i \in [m], j \in [n], \\
\frac{\partial g_{2,i}}{\partial w_j} &= 0 \text{ for any } i, j \in [m], \\
\frac{\partial g_{3,i}}{\partial x_i} &= 1 \text{ for all } i \in [n] \text{ and zero all the other partial derivatives of } g_{3,i}, \\
\frac{\partial g_{4,i}}{\partial y_i} &= 1 \text{ for all } i \in [m] \text{ and zero all the other partial derivatives of } g_{4,i}.
\end{aligned} \tag{13}$$

### A.2 Equations computed at point $(\mathbf{x}^*, \mathbf{y}^*, \mathbf{x}^*, \mathbf{y}^*)$

Set  $S_x = \sum_{t=1}^n x_t^* e^{\eta(A\mathbf{y}^*)_t}$ ,  $S_y = \sum_{t=1}^m y_t^* e^{-\eta(A^\top \mathbf{x}^*)_t}$  and let  $i, j$  be arbitrary indexes ( $g_{1,i}$  captures the  $i$ -th coordinate of function  $g_1$  etc). Assume  $\mathbf{v} = \mathbf{x}^{*\top} A \mathbf{y}^*$ , it is not hard to see

## 27:14 Last-Iterate Convergence

that  $(A^\top \mathbf{x}^*)_i = (A\mathbf{y}^*)_j = v$  for all  $i \in \text{Supp}(\mathbf{x}^*), j \in \text{Supp}(\mathbf{y}^*)$  and  $S_x = e^{\eta v}, S_y = e^{-\eta v}$ . We get that:

$$\begin{aligned}
\frac{\partial g_{1,i}}{\partial x_i} &= 1 - x_i^* \text{ for all } i \in \text{Supp}(\mathbf{x}^*), \\
\frac{\partial g_{1,i}}{\partial x_i} &= \frac{e^{\eta(A\mathbf{y}^*)_i}}{e^{\eta v}} \text{ for all } i \notin \text{Supp}(\mathbf{x}^*), \\
\frac{\partial g_{1,i}}{\partial x_j} &= -x_i^* \text{ for all } i, j \in \text{Supp}(\mathbf{x}^*) \text{ and } j \neq i, \\
\frac{\partial g_{1,i}}{\partial x_j} &= 0 \text{ for all } i \notin \text{Supp}(\mathbf{x}^*), j \in [n] \text{ and } j \neq i, \\
\frac{\partial g_{1,i}}{\partial y_j} &= x_i^*(2\eta A_{ij} - 2\eta v) \text{ for all } i \in \text{Supp}(\mathbf{x}^*), j \in \text{Supp}(\mathbf{y}^*), \\
\frac{\partial g_{1,i}}{\partial y_j} &= 0 \text{ for all } i \notin \text{Supp}(\mathbf{x}^*), j \in [m], \\
\frac{\partial g_{1,i}}{\partial z_j} &= 0 \text{ for all } i, j \in [n], \\
\frac{\partial g_{1,i}}{\partial w_j} &= x_i^*(-\eta A_{ij} + \eta v) \text{ for all } i \in \text{Supp}(\mathbf{x}^*), j \in \text{Supp}(\mathbf{y}^*), \\
\frac{\partial g_{1,i}}{\partial w_j} &= 0 \text{ for all } i \notin \text{Supp}(\mathbf{x}^*), j \in [m], \\
\frac{\partial g_{2,i}}{\partial y_i} &= 1 - y_i^* \text{ for all } i \in \text{Supp}(\mathbf{y}^*), \\
\frac{\partial g_{2,i}}{\partial y_i} &= \frac{e^{-\eta(A\mathbf{x}^*)_i}}{e^{-\eta v}} \text{ for all } i \notin \text{Supp}(\mathbf{y}^*), \\
\frac{\partial g_{2,i}}{\partial y_j} &= -y_i^* \text{ for all } i, j \in \text{Supp}(\mathbf{y}^*) \text{ and } j \neq i, \\
\frac{\partial g_{2,i}}{\partial y_j} &= 0 \text{ for all } i \notin \text{Supp}(\mathbf{y}^*), j \in [m] \text{ and } j \neq i, \\
\frac{\partial g_{2,i}}{\partial x_j} &= y_i^*(-2\eta A_{ij}^\top + 2\eta v) \text{ for all } i \in \text{Supp}(\mathbf{y}^*), j \in \text{Supp}(\mathbf{x}^*), \\
\frac{\partial g_{2,i}}{\partial x_j} &= 0 \text{ for all } i \notin \text{Supp}(\mathbf{y}^*), j \in [n], \\
\frac{\partial g_{2,i}}{\partial z_j} &= y_i^*(\eta A_{ij}^\top - \eta v) \text{ for all } i \in \text{Supp}(\mathbf{y}^*), j \in \text{Supp}(\mathbf{x}^*), \\
\frac{\partial g_{2,i}}{\partial z_j} &= 0 \text{ for all } i \notin \text{Supp}(\mathbf{y}^*), j \in [n], \\
\frac{\partial g_{2,i}}{\partial w_j} &= 0 \text{ for any } i, j \in [m], \\
\frac{\partial g_{3,i}}{\partial x_i} &= 1 \text{ for all } i \in [n] \text{ and zero all the other partial derivatives of } g_{3,i}, \\
\frac{\partial g_{4,i}}{\partial y_i} &= 1 \text{ for all } i \in [m] \text{ and zero all the other partial derivatives of } g_{4,i}.
\end{aligned} \tag{14}$$

### B Missing claims and proofs

Lemma 11 shows that the change between next and current iterate in both OMWU algorithms (classic and linear variant) is of order  $O(\eta)$  and that the difference between the next iterate of both algorithms is  $O(\eta^2)$ .

► **Lemma 11.** *Let  $\mathbf{x} \in \Delta_n$  be the vector of the max player,  $\mathbf{w}, \mathbf{z} \in \Delta_m$  and suppose  $\mathbf{x}', \mathbf{x}''$  are the next iterates of OMWU and its linear variant with current vector  $\mathbf{x}$  and vectors  $\mathbf{w}, \mathbf{z}$  of the min player. It holds that*

$$\|\mathbf{x}' - \mathbf{x}''\|_1 \text{ is } O(\eta^2), \text{ and } \|\mathbf{x}' - \mathbf{x}\|_1, \|\mathbf{x}'' - \mathbf{x}\|_1 \text{ are } O(\eta).$$

Analogously, it holds for vector  $\mathbf{y} \in \Delta_m$  of the min player and its next iterates.

**Proof.** Let  $\eta$  be sufficiently small (smaller than maximum in absolute value entry of  $A$ ).

$$\begin{aligned}
|x'_i - x''_i| &= x_i \left| \frac{e^{2\eta(A\mathbf{w})_i - \eta(A\mathbf{z})_i}}{\sum_j x_j e^{2\eta(A\mathbf{w})_j - \eta(A\mathbf{z})_j}} - \frac{1 + 2\eta(A\mathbf{w})_i - \eta(A\mathbf{z})_i}{\sum_j x_j (1 + 2\eta(A\mathbf{w})_j - \eta(A\mathbf{z})_j)} \right| \\
&= x_i \left| \frac{1 + 2\eta(A\mathbf{w})_i - \eta(A\mathbf{z})_i \pm O(\eta^2)}{\sum_j x_j (1 + 2\eta(A\mathbf{w})_j - \eta(A\mathbf{z})_j) \pm O(\eta^2)} - \frac{1 + 2\eta(A\mathbf{w})_i - \eta(A\mathbf{z})_i}{\sum_j x_j (1 + 2\eta(A\mathbf{w})_j - \eta(A\mathbf{z})_j)} \right| \\
&\text{which is } O(\eta^2)x_i
\end{aligned}$$



and hence  $\|\mathbf{x}' - \mathbf{x}''\|_1$  is  $O(\eta^2)$ . Moreover we have that

$$\begin{aligned} |x_i - x_i''| &= x_i \left| 1 - \frac{1 + 2\eta(\mathbf{A}\mathbf{w})_i - \eta(\mathbf{A}\mathbf{z})_i}{\sum_j x_j(1 + 2\eta(\mathbf{A}\mathbf{w})_j - \eta(\mathbf{A}\mathbf{z})_j)} \right| \\ &= x_i \left| \frac{\sum_j x_j(1 + 2\eta(\mathbf{A}\mathbf{w})_j - \eta(\mathbf{A}\mathbf{z})_j) - (1 + 2\eta(\mathbf{A}\mathbf{w})_i - \eta(\mathbf{A}\mathbf{z})_i)}{\sum_j x_j(1 + 2\eta(\mathbf{A}\mathbf{w})_j - \eta(\mathbf{A}\mathbf{z})_j)} \right| \\ &= x_i \left| \frac{\sum_j x_j(2\eta(\mathbf{A}\mathbf{w})_j - \eta(\mathbf{A}\mathbf{z})_j) - 2\eta(\mathbf{A}\mathbf{w})_i + \eta(\mathbf{A}\mathbf{z})_i}{\sum_j x_j(1 + 2\eta(\mathbf{A}\mathbf{w})_j - \eta(\mathbf{A}\mathbf{z})_j)} \right| \text{ which is } O(\eta)x_i. \end{aligned}$$

By triangle inequality and the two above proofs we get the third part of the lemma.  $\blacktriangleleft$

Lemmas 12, 13 and 15 will be used in the proof of Theorem 6.

**► Lemma 12.** *Let  $\mathbf{x} \in \Delta_n$ ,  $\mathbf{w}, \mathbf{z} \in \Delta_m$  and suppose  $\mathbf{x}', \mathbf{x}''$  are the next iterates of OMWU and its linear variant with current vector  $\mathbf{x}$  and inputs  $\mathbf{w}, \mathbf{z}$ , i.e.,  $\mathbf{x}'$  has coordinates  $x'_i = x_i \frac{e^{2\eta(\mathbf{A}\mathbf{w})_i - \eta(\mathbf{A}\mathbf{z})_i}}{\sum_j x_j e^{2\eta(\mathbf{A}\mathbf{w})_j - \eta(\mathbf{A}\mathbf{z})_j}}$  and  $\mathbf{x}''$  has coordinates  $x''_i = x_i \frac{1 + 2\eta(\mathbf{A}\mathbf{w})_i - \eta(\mathbf{A}\mathbf{z})_i}{\sum_j x_j(1 + 2\eta(\mathbf{A}\mathbf{w})_j - \eta(\mathbf{A}\mathbf{z})_j)}$ . It holds that (for  $\eta$  sufficiently small)*

$$\begin{aligned} \eta \mathbf{x}'^\top \mathbf{A}(2\mathbf{w} - \mathbf{z}) - \eta \mathbf{x}^\top \mathbf{A}(2\mathbf{w} - \mathbf{z}) &= \\ &= (\eta \mathbf{x}''^\top \mathbf{A}(2\mathbf{w} - \mathbf{z}) - \eta \mathbf{x}^\top \mathbf{A}(2\mathbf{w} - \mathbf{z})) - O(\eta^3) = \\ &= (1 - O(\eta))\eta^2 \sum_i x_i (2\mathbf{x}^\top \mathbf{A}\mathbf{w} - \mathbf{x}^\top \mathbf{A}\mathbf{z} - 2(\mathbf{A}\mathbf{w})_i + (\mathbf{A}\mathbf{z})_i)^2 - O(\eta^3) \\ &= (1 - O(\eta))\eta^2 \sum_i x'_i (2\mathbf{x}'^\top \mathbf{A}\mathbf{w} - \mathbf{x}'^\top \mathbf{A}\mathbf{z} - 2(\mathbf{A}\mathbf{w})_i + (\mathbf{A}\mathbf{z})_i)^2 - O(\eta^3). \end{aligned}$$

**Proof.** It suffices to prove the second equality. The rest follow from Lemma 11. Set  $B = (\mathbf{1}_n \mathbf{1}_m^\top + \eta A)$ . We have that  $x''_i = x_i \frac{(B(2\mathbf{w} - \mathbf{z}))_i}{\mathbf{x}^\top B(2\mathbf{w} - \mathbf{z})}$  (from definition of linear variant of OMWU dynamics). It follows that

$$\begin{aligned} (\mathbf{x}''^\top B(2\mathbf{w} - \mathbf{z})) \cdot (\mathbf{x}^\top B(2\mathbf{w} - \mathbf{z})) &= \sum_{ij} B_{ij} x''_i (2\mathbf{w} - \mathbf{z})_j \cdot (\mathbf{x}^\top B(2\mathbf{w} - \mathbf{z})) \\ &= \sum_{ij} B_{ij} \left( x_i \frac{(B(2\mathbf{w} - \mathbf{z}))_i}{\mathbf{x}^\top B(2\mathbf{w} - \mathbf{z})} \right) (2\mathbf{w} - \mathbf{z})_j \cdot (\mathbf{x}^\top B(2\mathbf{w} - \mathbf{z})) \\ &= \sum_{ij} B_{ij} (x_i (B(2\mathbf{w} - \mathbf{z}))_i) (2\mathbf{w} - \mathbf{z})_j \\ &= \sum_i x_i (B(2\mathbf{w} - \mathbf{z}))_i^2 \\ &= (\mathbf{x}^\top B(2\mathbf{w} - \mathbf{z}))^2 + \sum_i x_i (\mathbf{x}^\top B(2\mathbf{w} - \mathbf{z}) - (B(2\mathbf{w} - \mathbf{z}))_i)^2. \end{aligned}$$

where the last inequality comes from the fact that for a random variable  $\xi$  we have  $\mathbb{E}[\xi^2] = \mathbb{E}^2[\xi] + \mathbb{V}[\xi]$ . Therefore by dividing LHS and RHS by  $\mathbf{x}^\top B(2\mathbf{w} - \mathbf{z})$  which is  $1 \pm O(\eta)$  we get

$$\begin{aligned} (\mathbf{x}''^\top B(2\mathbf{w} - \mathbf{z})) &= (\mathbf{x}^\top B(2\mathbf{w} - \mathbf{z})) + (1 - O(\eta)) \sum_i x_i (\mathbf{x}^\top B(2\mathbf{w} - \mathbf{z}) - (B(2\mathbf{w} - \mathbf{z}))_i)^2 \\ &= (\mathbf{x}^\top (\mathbf{1}_n \mathbf{1}_m^\top + \eta A)(2\mathbf{w} - \mathbf{z})) + \\ &\quad + (1 - O(\eta))\eta^2 \sum_i x_i (\mathbf{x}^\top \mathbf{A}(2\mathbf{w} - \mathbf{z}) - (\mathbf{A}(2\mathbf{w} - \mathbf{z}))_i)^2. \end{aligned}$$

The proof is complete by Lemma 11.  $\blacktriangleleft$

## 27:16 Last-Iterate Convergence

Using same arguments as in proof of Lemma 12 we have the following lemma:

► **Lemma 13.** Let  $\mathbf{y} \in \Delta_m$ ,  $\mathbf{w}, \mathbf{z} \in \Delta_n$  and suppose  $\mathbf{y}'$  is the next iterate of OMWU with current vector  $\mathbf{y}$  and inputs  $\mathbf{w}, \mathbf{z}$ , i.e.,  $\mathbf{y}'$  has coordinates  $y'_i = y_i \frac{e^{-2\eta(A^\top \mathbf{w})_i + \eta(A^\top \mathbf{z})_i}}{\sum_j y_j e^{-2\eta(A^\top \mathbf{w})_j + \eta(A^\top \mathbf{z})_j}}$ . It holds that (for  $\eta$  sufficiently small)

$$\begin{aligned} & \eta \mathbf{y}'^\top A^\top (\mathbf{z} - 2\mathbf{w}) - \eta \mathbf{y}^\top A^\top (\mathbf{z} - 2\mathbf{w}) \\ &= (1 - O(\eta)) \eta^2 \sum_i y'_i (\mathbf{y}'^\top A^\top \mathbf{z} - 2\mathbf{y}'^\top A^\top \mathbf{w} - (A^\top \mathbf{z})_i + 2(A^\top \mathbf{w})_i)^2 - O(\eta^3). \end{aligned}$$

► **Lemma 14.** Let  $(\mathbf{x}^t, \mathbf{y}^t)$  be the  $t$ -th iterate of OMWU dynamics (5). For each time step  $t \geq 2$  it holds that

$$\begin{aligned} & \eta \mathbf{x}^{t-1 \top} A \mathbf{y}^t - \eta \mathbf{x}^t \top A \mathbf{y}^{t-1} \leq \\ & \leq -(1 - O(\eta)) \eta^2 \sum_i x_i^t (2\mathbf{x}^{t \top} A \mathbf{y}^t - \mathbf{x}^t \top A \mathbf{y}^{t-1} - 2(A \mathbf{y}^t)_i + (A \mathbf{y}^{t-1})_i)^2 - \\ & - (1 - O(\eta)) \eta^2 \sum_i y_i^t (\mathbf{y}^{t \top} A^\top \mathbf{x}^{t-1} - 2\mathbf{y}^{t \top} A^\top \mathbf{x}^t - (A^\top \mathbf{x}^{t-1})_i + 2(A^\top \mathbf{x}^t)_i)^2 + O(\eta^3). \end{aligned}$$

**Proof.**

$$\begin{aligned} & \eta \mathbf{x}^{t-1 \top} A \mathbf{y}^t - \eta \mathbf{x}^t \top A \mathbf{y}^{t-1} = \\ & = \eta \mathbf{x}^{t-1 \top} A \mathbf{y}^t - \frac{1}{2} \eta \mathbf{x}^{t-1 \top} A \mathbf{y}^{t-1} + \frac{1}{2} \eta \mathbf{x}^{t-1 \top} A \mathbf{y}^{t-1} - \eta \mathbf{x}^t \top A \mathbf{y}^{t-1} \\ & = \eta \mathbf{x}^t \top A \mathbf{y}^t - \frac{1}{2} \eta \mathbf{x}^t \top A \mathbf{y}^{t-1} + \frac{1}{2} \eta \mathbf{x}^{t-1 \top} A \mathbf{y}^t - \eta \mathbf{x}^t \top A \mathbf{y}^t - \\ & - \left( \frac{1}{2} - O(\eta) \right) \eta^2 \sum_i x_i^t (2\mathbf{x}^{t \top} A \mathbf{y}^t - \mathbf{x}^t \top A \mathbf{y}^{t-1} - 2(A \mathbf{y}^t)_i + (A \mathbf{y}^{t-1})_i)^2 - \\ & - \left( \frac{1}{2} - O(\eta) \right) \eta^2 \sum_i y_i^t (\mathbf{y}^{t \top} A^\top \mathbf{x}^{t-1} - 2\mathbf{y}^{t \top} A^\top \mathbf{x}^t - (A^\top \mathbf{x}^{t-1})_i + \\ & + 2(A^\top \mathbf{x}^t)_i)^2 + O(\eta^3). \end{aligned}$$

The second equality comes from Lemmas 12 and 13. By canceling out the common terms and bring to the LHS the appropriate remaining terms, the claim follows. ◀

► **Lemma 15.** Let  $(\mathbf{x}^t, \mathbf{y}^t)$  denote the  $t$ -th iterate of OMWU dynamics. It holds for  $t \geq 2$  that

$$\mathbf{x}^* \top A (2\mathbf{y}^t - \mathbf{y}^{t-1}) \geq \mathbf{x}^* \top A \mathbf{y}^* \text{ and } (2\mathbf{x}^t \top - \mathbf{x}^{t-1 \top}) A \mathbf{y}^* \leq \mathbf{x}^* \top A \mathbf{y}^*,$$

where  $(\mathbf{x}^*, \mathbf{y}^*)$  is the optimal solution of the min-max problem.

**Proof.** It is true that  $x_i^t \geq (1 - O(\eta)) x_i^{t-1}$ , hence  $x_i^t \geq \frac{1}{2} x_i^{t-1}$  for  $\eta$  sufficiently small. Therefore  $2\mathbf{x}^t - \mathbf{x}^{t-1}$  lies in the simplex  $\Delta_n$ . Hence since  $(\mathbf{x}^*, \mathbf{y}^*)$  is the optimum (Nash equilibrium) we get that  $(2\mathbf{x}^t \top - \mathbf{x}^{t-1 \top}) A \mathbf{y}^* \leq \mathbf{x}^* \top A \mathbf{y}^*$  ( $\mathbf{x}$  is the max player). Similarly the second inequality can be proved. ◀

**Proof of Theorem 6.** We compute the difference between

$D_{KL}((\mathbf{x}^*, \mathbf{y}^*) || (\mathbf{x}^{t+1}, \mathbf{y}^{t+1}))$  and  $D_{KL}((\mathbf{x}^*, \mathbf{y}^*) || (\mathbf{x}^t, \mathbf{y}^t))$

$$\begin{aligned}
& D_{KL}((\mathbf{x}^*, \mathbf{y}^*) || (\mathbf{x}^{t+1}, \mathbf{y}^{t+1})) - D_{KL}((\mathbf{x}^*, \mathbf{y}^*) || (\mathbf{x}^t, \mathbf{y}^t)) \\
&= - \left( \sum_i \mathbf{x}_i^* \ln \frac{x_i^{t+1}}{x_i^t} + \sum_i y_i^* \ln \frac{y_i^{t+1}}{y_i^t} \right) \\
&= - \left( \sum_i \mathbf{x}_i^* \ln e^{2\eta(\mathbf{A}\mathbf{y}^t)_i - \eta(\mathbf{A}\mathbf{y}^{t-1})_i} + \sum_i y_i^* \ln e^{-2\eta(\mathbf{A}^\top \mathbf{x}^t)_i + \eta(\mathbf{A}^\top \mathbf{x}^{t-1})_i} \right) + \\
&\quad + \ln \left( \sum_i x_i^t e^{2\eta(\mathbf{A}\mathbf{y}^t)_i - \eta(\mathbf{A}\mathbf{y}^{t-1})_i} \right) + \ln \left( \sum_i y_i^t e^{-2\eta(\mathbf{A}^\top \mathbf{x}^t)_i + \eta(\mathbf{A}^\top \mathbf{x}^{t-1})_i} \right) \\
&= -2\eta \mathbf{x}^{* \top} \mathbf{A}\mathbf{y}^t + \eta \mathbf{x}^{* \top} \mathbf{A}\mathbf{y}^{t-1} + 2\eta \mathbf{x}^t \top \mathbf{A}\mathbf{y}^* - \eta \mathbf{x}^{t-1 \top} \mathbf{A}\mathbf{y}^* + \\
&\quad + \ln \left( \sum_i x_i^t e^{2\eta(\mathbf{A}\mathbf{y}^t)_i - \eta(\mathbf{A}\mathbf{y}^{t-1})_i} \right) + \ln \left( \sum_i y_i^t e^{-2\eta(\mathbf{A}^\top \mathbf{x}^t)_i + \eta(\mathbf{A}^\top \mathbf{x}^{t-1})_i} \right).
\end{aligned}$$

We use Lemma 15 and get that  $-2\eta \mathbf{x}^{* \top} \mathbf{A}\mathbf{y}^t + \eta \mathbf{x}^{* \top} \mathbf{A}\mathbf{y}^{t-1} + 2\eta \mathbf{x}^t \top \mathbf{A}\mathbf{y}^* - \eta \mathbf{x}^{t-1 \top} \mathbf{A}\mathbf{y}^* \leq 0$ , therefore the LHS (difference in the KL divergence) is at most

$$\begin{aligned}
& \overbrace{-2\eta \mathbf{x}^t \top \mathbf{A}\mathbf{y}^t + \eta \mathbf{x}^t \top \mathbf{A}\mathbf{y}^{t-1} + 2\eta \mathbf{x}^t \top \mathbf{A}\mathbf{y}^* - \eta \mathbf{x}^{t-1 \top} \mathbf{A}\mathbf{y}^t - \eta \mathbf{x}^t \top \mathbf{A}\mathbf{y}^{t-1} + \eta \mathbf{x}^{t-1 \top} \mathbf{A}\mathbf{y}^t}_{=0} \\
&\quad + \ln \left( \sum_i x_i^t e^{2\eta(\mathbf{A}\mathbf{y}^t)_i - \eta(\mathbf{A}\mathbf{y}^{t-1})_i} \right) + \ln \left( \sum_i y_i^t e^{-2\eta(\mathbf{A}^\top \mathbf{x}^t)_i + \eta(\mathbf{A}^\top \mathbf{x}^{t-1})_i} \right) \\
&= \ln \left( \sum_i x_i^t e^{2\eta((\mathbf{A}\mathbf{y}^t)_i - \mathbf{x}^t \top \mathbf{A}\mathbf{y}^t) - \eta((\mathbf{A}\mathbf{y}^{t-1})_i - \mathbf{x}^t \top \mathbf{A}\mathbf{y}^{t-1})} \right) + \\
&\quad \ln \left( \sum_i y_i^t e^{-2\eta((\mathbf{A}^\top \mathbf{x}^t)_i - \mathbf{x}^t \top \mathbf{A}\mathbf{y}^t) + \eta((\mathbf{A}^\top \mathbf{x}^{t-1})_i - \mathbf{x}^{t-1 \top} \mathbf{A}\mathbf{y}^t)} \right) - \eta \mathbf{x}^t \top \mathbf{A}\mathbf{y}^{t-1} + \eta \mathbf{x}^{t-1 \top} \mathbf{A}\mathbf{y}^t
\end{aligned}$$

We furthermore use second order Taylor approximation ( $\eta$  is sufficiently small) to the function  $e^x$  and we get that previous expression is at most

$$\begin{aligned}
& \leq \ln \left( \sum_i x_i^t (1 + 2\eta((\mathbf{A}\mathbf{y}^t)_i - \mathbf{x}^t \top \mathbf{A}\mathbf{y}^t) - \eta((\mathbf{A}\mathbf{y}^{t-1})_i - \mathbf{x}^t \top \mathbf{A}\mathbf{y}^{t-1})) + \right. \\
&\quad \left. + \sum_i x_i^t \left( \frac{1}{2} + O(\eta) \right) \eta^2 (2(\mathbf{A}\mathbf{y}^t)_i - 2\mathbf{x}^t \top \mathbf{A}\mathbf{y}^t - (\mathbf{A}\mathbf{y}^{t-1})_i + \mathbf{x}^t \top \mathbf{A}\mathbf{y}^{t-1})^2 \right) + \\
&\quad + \ln \left( \sum_i y_i^t (1 - 2\eta((\mathbf{A}^\top \mathbf{x}^t)_i - \mathbf{x}^t \top \mathbf{A}\mathbf{y}^t) + \eta((\mathbf{A}^\top \mathbf{x}^{t-1})_i - \mathbf{x}^{t-1 \top} \mathbf{A}\mathbf{y}^t)) + \right. \\
&\quad \left. + \sum_i y_i^t \left( \frac{1}{2} + O(\eta) \right) \eta^2 (2(\mathbf{A}^\top \mathbf{x}^t)_i - 2\mathbf{x}^t \top \mathbf{A}\mathbf{y}^t - (\mathbf{A}^\top \mathbf{x}^{t-1})_i + \mathbf{x}^{t-1 \top} \mathbf{A}\mathbf{y}^t)^2 \right) - \\
&\quad - \eta \mathbf{x}^t \top \mathbf{A}\mathbf{y}^{t-1} + \eta \mathbf{x}^{t-1 \top} \mathbf{A}\mathbf{y}^t
\end{aligned}$$

Finally, using Taylor approximation on  $\log(1+x)$  and Lemma 14 (last equality) we get

## 27:18 Last-Iterate Convergence

the following system:

$$\begin{aligned}
&\leq \eta \mathbf{x}^{t-1 \top} \mathbf{A} \mathbf{y}^t - \eta \mathbf{x}^t \top \mathbf{A} \mathbf{y}^{t-1} + \\
&+ \sum_i x_i^t \left( \left( \frac{1}{2} + O(\eta) \right) \eta^2 \left( 2(\mathbf{A} \mathbf{y}^t)_i - 2 \mathbf{x}^t \top \mathbf{A} \mathbf{y}^t - (\mathbf{A} \mathbf{y}^{t-1})_i + \mathbf{x}^t \top \mathbf{A} \mathbf{y}^{t-1} \right)^2 \right) + \\
&+ \sum_i y_i^t \left( \left( \frac{1}{2} + O(\eta) \right) \eta^2 \left( 2(\mathbf{A}^\top \mathbf{x}^t)_i - 2 \mathbf{x}^t \top \mathbf{A} \mathbf{y}^t - (\mathbf{A}^\top \mathbf{x}^{t-1})_i + \mathbf{x}^{t-1 \top} \mathbf{A} \mathbf{y}^t \right)^2 \right) \\
&= - \sum_i x_i^t \left( \left( \frac{1}{2} - O(\eta) \right) \eta^2 \left( 2(\mathbf{A} \mathbf{y}^t)_i - 2 \mathbf{x}^t \top \mathbf{A} \mathbf{y}^t - (\mathbf{A} \mathbf{y}^{t-1})_i + \mathbf{x}^t \top \mathbf{A} \mathbf{y}^{t-1} \right)^2 \right) \\
&- \sum_i y_i^t \left( \left( \frac{1}{2} - O(\eta) \right) \eta^2 \left( 2(\mathbf{A}^\top \mathbf{x}^t)_i - 2 \mathbf{x}^t \top \mathbf{A} \mathbf{y}^t - (\mathbf{A}^\top \mathbf{x}^{t-1})_i + \mathbf{x}^{t-1 \top} \mathbf{A} \mathbf{y}^t \right)^2 \right) + O(\eta^3).
\end{aligned}$$

It is clear that as long as  $(\mathbf{x}^t, \mathbf{y}^t)$  (and thus  $(\mathbf{x}^{t-1}, \mathbf{y}^{t-1})$  by Lemma 11) is not  $O(\eta^{1/3})$ -close, from above inequalities/equalities we get

$$D_{KL}((\mathbf{x}^*, \mathbf{y}^*) || (\mathbf{x}^{t+1}, \mathbf{y}^{t+1})) - D_{KL}((\mathbf{x}^*, \mathbf{y}^*) || (\mathbf{x}^t, \mathbf{y}^t)) \leq -\Omega(\eta^3),$$

meaning that KL divergence decreases by at least a factor of  $\eta^3$  and the claim follows. ◀

► **Lemma 16.** *Let  $D$  be a real diagonal matrix with positive diagonal entries and  $S$  be a real skew-symmetric matrix ( $S^\top = -S$ ). It holds that  $SD$  has eigenvalues with real part zero (i.e., it has only imaginary eigenvalues).*

**Proof.** Let  $\mathbf{z}^*$  be the conjugate transpose of  $\mathbf{z}$  and  $\mathbf{z}^*$  be a left eigenvector of  $SD$  with complex eigenvalue  $\lambda$ . It holds that

$$\begin{aligned}
\lambda \mathbf{z}^* D^{-1} \mathbf{z} &= \mathbf{z}^* S D D^{-1} \mathbf{z} \\
&= \mathbf{z}^* S \mathbf{z} \\
&= -(\mathbf{z}^* S \mathbf{z})^* \text{ (since } S \text{ is skew symmetric)} \\
&= -(\lambda \mathbf{z}^* D^{-1} \mathbf{z})^* \text{ (using first and second equalities above)} \\
&= -\bar{\lambda} \mathbf{z}^* D^{-1} \mathbf{z}.
\end{aligned}$$

Since  $D$  has positive diagonal entries, we conclude that  $\mathbf{z}^* D^{-1} \mathbf{z} \neq 0$  (since  $\mathbf{z} \neq \mathbf{0}$ ), thus  $\lambda = -\bar{\lambda}$  and the claim follows. ◀

# Density Estimation for Shift-Invariant Multidimensional Distributions

Anindya De<sup>1</sup>

EECS Department, Northwestern University, 2145 Sheridan Road, Evanston, IL 60208, USA  
anindya@eecs.northwestern.edu

Philip M. Long

Google, 1600 Amphitheatre Parkway, Mountain View, CA 94043, USA  
plong@google.com

Rocco A. Servedio<sup>2</sup>

Department of Computer Science, Columbia University, 500 W. 120th Street, Room 450,  
New York, NY 10027, USA  
rocco@cs.columbia.edu

---

## Abstract

---

We study density estimation for classes of *shift-invariant* distributions over  $\mathbb{R}^d$ . A multidimensional distribution is “shift-invariant” if, roughly speaking, it is close in total variation distance to a small shift of it in any direction. Shift-invariance relaxes smoothness assumptions commonly used in non-parametric density estimation to allow jump discontinuities. The different classes of distributions that we consider correspond to different rates of tail decay.

For each such class we give an efficient algorithm that learns any distribution in the class from independent samples with respect to total variation distance. As a special case of our general result, we show that  $d$ -dimensional shift-invariant distributions which satisfy an exponential tail bound can be learned to total variation distance error  $\varepsilon$  using  $\tilde{O}_d(1/\varepsilon^{d+2})$  examples and  $\tilde{O}_d(1/\varepsilon^{2d+2})$  time. This implies that, for constant  $d$ , multivariate log-concave distributions can be learned in  $\tilde{O}_d(1/\varepsilon^{2d+2})$  time using  $\tilde{O}_d(1/\varepsilon^{d+2})$  samples, answering a question of [29]. All of our results extend to a model of *noise-tolerant* density estimation using Huber’s contamination model, in which the target distribution to be learned is a  $(1 - \varepsilon, \varepsilon)$  mixture of some unknown distribution in the class with some other arbitrary and unknown distribution, and the learning algorithm must output a hypothesis distribution with total variation distance error  $O(\varepsilon)$  from the target distribution. We show that our general results are close to best possible by proving a simple  $\Omega(1/\varepsilon^d)$  information-theoretic lower bound on sample complexity even for learning bounded distributions that are shift-invariant.

**2012 ACM Subject Classification** Theory of computation → Unsupervised learning and clustering

**Keywords and phrases** Density estimation, unsupervised learning, log-concave distributions, non-parametrics

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2019.28

**Related Version** A full version of the paper is available at [20], <https://arxiv.org/abs/1811.03744>.

---

<sup>1</sup> Supported by NSF grant CCF-1814706.

<sup>2</sup> Supported by NSF grants CCF-1563155, IIS-1838154 and CCF-1814873.



© Anindya De, Philip M. Long, and Rocco A. Servedio;  
licensed under Creative Commons License CC-BY

10th Innovations in Theoretical Computer Science (ITCS 2019).

Editor: Avrim Blum; Article No. 28; pp. 28:1–28:20



Leibniz International Proceedings in Informatics

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

## 1 Introduction

In multidimensional density estimation, an algorithm has access to independent draws from an unknown target probability distribution over  $\mathbb{R}^d$ , which is typically assumed to belong to or be close to some class of “nice” distributions. The goal is to output a hypothesis distribution which with high probability is close to the target distribution. A number of different distance measures can be used to capture the notion of closeness; in this work we use the total variation distance (also known as the “statistical distance” and equivalent to the  $L_1$  distance). This is a well studied framework which has been investigated in detail, see e.g. the books [23, 24].

Multidimensional density estimation is typically attacked in one of two ways. In the first general approach a parameterized hypothesis class is chosen, and a setting of parameters is chosen based on the observed data points. This approach is justified given the belief that the parameterized class contains a good approximation to the distribution generating the data, or even that the parameterized class actually contains the target distribution. See [14, 33, 39] for some well-known multidimensional distribution learning results in this line.

In the second general approach a hypothesis distribution is constructed by “smoothing” the empirical distribution with a kernel function. This approach is justified by the belief that the target distribution satisfies some smoothness assumptions, and is more appropriate when studying distributions that do not have a parametric representation. The current paper falls within this second strand.

The most popular smoothness assumption is that the distribution has a density that belongs to a Sobolev space [42, 6, 30, 24]. The simplest Sobolev space used in this context corresponds to having a bound on the average of the partial first “weak derivatives” of the density; other Sobolev spaces correspond to bounding additional derivatives. A drawback of this approach is that it does not apply to distributions whose densities have jump discontinuities. Such jump discontinuities can arise in various applications, for example, when objects under analysis must satisfy hard constraints.

To address this, some authors have used the weaker assumption that the density belongs to a Besov space [7, 22, 38, 43, 3]. In the simplest case, this allows jump discontinuities as long as the function does not change very fast on average. The precise definition, which is quite technical (see [22]), makes reference to the effect on a distribution of shifting the domain by a small amount.

**The densities we consider.** In this paper we analyze a clean and simple smoothness assumption, which is a continuous analog of the notion of shift-invariance that has recently been used for analyzing the learnability of various types of discrete distributions [5, 16, 21]. The assumption is based on the *shift-invariance of  $f$  in direction  $v$  at scale  $\kappa$* , which, for a density  $f$  over  $\mathbb{R}^d$ , a unit vector  $v \in \mathbb{R}^d$ , and a positive real value  $\kappa$ , we define to be

$$\text{SI}(f, v, \kappa) \stackrel{\text{def}}{=} \frac{1}{\kappa} \cdot \sup_{\kappa' \in [0, \kappa]} \int_{\mathbb{R}^d} |f(x + \kappa'v) - f(x)| dx.$$

We define the quantity  $\text{SI}(f, \kappa)$  to be the worst case of  $\text{SI}(f, v, \kappa)$  over all directions  $v$ , i.e.  $\text{SI}(f, \kappa) \stackrel{\text{def}}{=} \sup_{v: \|v\|_2=1} \text{SI}(f, v, \kappa)$ . For any constant  $c$ , we define the class of densities  $\mathcal{C}_{\text{SI}}(c, d)$  to consist of all  $d$ -dimensional densities  $f$  with the property that  $\text{SI}(f, \kappa) \leq c$  for all  $\kappa > 0$ .

Our notion of shift-invariance provides a quantitative way of capturing the intuition that the density  $f$  changes gradually on average in every direction. Several natural classes fit nicely into this framework; for example, we note that  $d$ -dimensional standard normal distributions are easily shown to belong to  $\mathcal{C}_{\text{SI}}(1, d)$ . As another example, we will show later that any  $d$ -dimensional isotropic log-concave distribution belongs to  $\mathcal{C}_{\text{SI}}(O_d(1), d)$ .

Many distributions arising in practice have light tails, and distributions with light tails can in general be learned more efficiently. To analyze learning shift-invariant distributions in a manner that takes advantage of light tails when they are available, while accommodating heavier tails when necessary, we define classes with different combinations of shift-invariant and tail behavior. Given a nonincreasing function  $g : \mathbb{R}^+ \rightarrow [0, 1]$  which satisfies  $\lim_{t \rightarrow +\infty} g(t) = 0$ , we define the class of densities  $\mathcal{C}_{\text{SI}}(c, d, g)$  to consist of those  $f \in \mathcal{C}_{\text{SI}}(c, d)$  which have the additional property that for all  $t > 0$ , it holds that  $\Pr_{\mathbf{x} \leftarrow f} [\|\mathbf{x} - \mu\| > t] \leq g(t)$ , where  $\mu \in \mathbb{R}^d$  is the mean of the distribution  $f$ .

As motivation for its study, we feel that  $\mathcal{C}_{\text{SI}}(c, d, g)$  is a simple and easily understood class that exhibits an attractive tradeoff between expressiveness and tractability. As we show, it is broad enough to include distributions of central interest such as multidimensional isotropic log-concave distributions, but it is also limited enough to admit efficient density estimation algorithms.

**Our density estimation framework.** We recall the standard notion of density estimation with respect to total variation distance. Given a class  $\mathcal{C}$  of densities over  $\mathbb{R}^d$ , a density estimation algorithm for  $\mathcal{C}$  is given access to i.i.d. draws from  $f$ , where  $f \in \mathcal{C}$  is the unknown *target density* to be learned. For any  $f \in \mathcal{C}$ , given any parameter  $\varepsilon > 0$ , after making some number of draws depending on  $d$  and  $\varepsilon$  the density estimation algorithm must output a description of a hypothesis density  $h$  over  $\mathbb{R}^d$  which, with high probability over the draws from  $f$ , satisfies  $d_{\text{TV}}(f, h) \leq \varepsilon$ . It is of interest both to bound the *sample complexity* of such an algorithm (the number of draws from  $f$  that it makes) and its running time. In the full version of this paper [20], we show that our learning results can be extended to a challenging model of *noise-tolerant* density estimation for a class  $\mathcal{C}$ .

## 1.1 Results

Our main positive result is a general algorithm which efficiently learns any class  $\mathcal{C}_{\text{SI}}(c, d, g)$ . Given a constant  $c$  and a tail bound  $g$ , we show that any distribution in the class  $\mathcal{C}_{\text{SI}}(c, d, g)$  can be learned to any error  $O(\varepsilon)$  with a sample complexity that depends on  $c, g, \varepsilon$  and  $d$ . The running time of our algorithm is roughly quadratic in the sample complexity, and the sample complexity is  $O_{c,d,g}(1) \cdot (\frac{1}{\varepsilon})^{d+2}$  (see Theorem 12 in Section 4 for a precise statement of the exact bound). These bounds on the number of examples and running time do not depend on which member of  $\mathcal{C}_{\text{SI}}(c, d, g)$  is being learned.

**Application: Learning multivariate log-concave densities.** A multivariate density function  $f$  over  $\mathbb{R}^d$  is said to be *log-concave* if there is an upper semi-continuous concave function  $\phi : \mathbb{R}^d \rightarrow [-\infty, \infty)$  such that  $f(x) = e^{\phi(x)}$  for all  $x$ . Log-concave distributions arise in a range of contexts and have been well studied; see [11, 12, 3, 1, 9, 25] for work on density estimation of univariate (discrete and continuous) log-concave distributions. In the multivariate case, [35] gave a sample complexity lower bound (for squared Hellinger distance) which implies that  $\Omega(1/\varepsilon^{(d+1)/2})$  samples are needed to learn  $d$ -dimensional log-concave densities to error  $\varepsilon$ . More recently, [29] established the first finite sample complexity upper bound for multivariate log-concave densities, by giving an algorithm that learns any  $d$ -dimensional log-concave density using  $\tilde{O}_d(1/\varepsilon^{(d+5)/2})$  samples. The algorithm of [29] is not computationally efficient, and indeed, Diakonikolas et al. ask if there is an algorithm with running time polynomial in the sample complexity, referring to this as “a challenging and important open question.” A subsequent (and recent) work of Carpenter et al. [10] showed that the maximum likelihood estimator (MLE) is statistically efficient (i.e., achieves near optimal sample complexity).



We show that multivariate log-concave densities can be learned in polynomial time as a special case of our main algorithmic result. We establish that any  $d$ -dimensional near-isotropic log-concave density is  $O_d(1)$ -shift-invariant. Together with well-known tail bounds on  $d$ -dimensional log-concave densities, this easily yields that any  $d$ -dimensional near-isotropic log-concave density belongs to  $\mathcal{C}_{\text{SI}}(c, d, g)$  where the tail bound function  $g$  is inverse exponential. Theorem 12 then immediately gives a  $\tilde{O}_d(1/\varepsilon^{2d+2})$ -time algorithm for learning near-isotropic log-concave densities. Adding a preprocessing step to reduce to the near-isotropic case yields an algorithm that works for all log-concave densities.

While our sample complexity is quadratically larger than the optimal sample complexity for learning log-concave distributions (from [29]), such *computational-statistical* tradeoffs are in fact quite common (see, for example, the work of [8] which gives a faster algorithm for learning Gaussian mixture models by using more samples).

**A lower bound.** We also prove a simple lower bound, showing that any algorithm that learns shift-invariant  $d$ -dimensional densities with bounded support to error  $\varepsilon$  must use  $\Omega(1/\varepsilon^d)$  examples. These densities may be thought of as satisfying the strongest possible rate of tail decay as they have zero tail mass outside of a bounded region (corresponding to  $g(t) = 0$  for  $t$  larger than some absolute constant). This lower bound shows that a sample complexity of at least  $1/\varepsilon^d$  is necessary even for very structured special cases of our multivariate density estimation problem.

## 1.2 Our approach

For simplicity, and because it is a key component of our general algorithm, we first describe how our algorithm learns an  $\varepsilon$ -error hypothesis when the target distribution belongs to  $\mathcal{C}_{\text{SI}}(c, d)$  and also has *bounded support*: all its mass is on points in the origin-centered ball of radius  $1/2$ .

In this special case, analyzed in Section 3, our algorithm has two conceptual stages. First, we smooth the density that we are to learn through convolution – this is done in a simple way by randomly perturbing each draw. This convolution uses a kernel that damps the contributions to the density coming from high-frequency functions in its Fourier decomposition; intuitively, the shift-invariance of the target density ensures that the convolved density (which is an average over small shifts of the original density) is close to the original density. In the second conceptual stage, the algorithm approximates relatively few Fourier coefficients of the smoothed density. We show that an inverse Fourier transformation using this approximation still provides an accurate approximation to the target density.<sup>3</sup>

Next, in Section 4, we consider the more general case in which the target distribution belongs to the class  $\mathcal{C}_{\text{SI}}(c, d, g)$ . Here the high-level idea of our approach is very straightforward: it is essentially to reduce to the simpler special case (of bounded support and good shift-invariance in every direction) described above. (A crucial aspect of this transformation algorithm is that it uses only a small number of draws from the original shift-invariant distribution; we return to this point below.) We can then use the algorithm for the special case to obtain a high-accuracy hypothesis, and perform the inverse transformation to obtain

---

<sup>3</sup> We note that a simpler version of this approach, which only uses a smoothing kernel and does not employ Fourier analysis, can be shown to give a similar, but quantitatively worse, results, such as a sample complexity of essentially  $1/\varepsilon^{2d}$  when  $g(t)$  is zero outside of a bounded region. However, this is worse than the lower bound of  $\Omega(1/\varepsilon^d)$  by a quadratic factor, whereas our algorithm essentially achieves this optimal sample complexity.

a high-accuracy hypothesis for the original general distribution. We remark that while the conceptual idea is thus very straightforward, there are a number of technical challenges that must be met to implement this approach. One of these is that it is necessary to truncate the tails of the original distribution so that an affine transformation of it will have bounded support, and doing this changes the shift-invariance of the original distribution. Another is that the transformation procedure only succeeds with non-negligible probability, so we must run this overall approach multiple times and perform hypothesis selection to actually end up with a single high-accuracy hypothesis.

In Section 5 we apply the above results to establish efficient learnability of log-concave densities over  $\mathbb{R}^d$ . To apply our results, we need to have (i) bounds on the rate of tail decay, and (ii) shift-invariance bounds. As noted earlier, exponential tail bounds on  $d$ -dimensional log-concave densities are well known, so it remains to establish shift-invariance. Using basic properties of log-concave densities, in Section 5 we show that any  $d$ -dimensional isotropic log-concave density is  $O_d(1)$ -shift-invariant. Armed with this bound, by applying our learning result (Theorem 12) we get that any  $d$ -dimensional isotropic log-concave density can be learned in time  $\tilde{O}_d(1/\varepsilon^{2d+2})$ , using  $\tilde{O}_d(1/\varepsilon^{d+2})$  samples. Log-concave distributions are shift-invariant even if they are only approximately isotropic. We show that general log-concave distributions may be learned by bringing them into approximately isotropic position with a preprocessing step, borrowing techniques from [37].

### 1.3 Related work

The most closely related work that we are aware of was mentioned above: Holmström and Klemelä [30] obtained bounds similar to ours for using kernel methods to learn densities that belong to various Sobolev spaces. As mentioned above, these results do not directly apply for learning densities in  $\mathcal{C}_{\text{SI}}(c, d, g)$  because of the possibility of jump discontinuities. Holmström and Klemelä also proved a lower bound on the sample complexity of algorithms that compute kernel density estimates. In contrast our lower bound holds for any density estimation algorithm, kernel-based or otherwise.

The assumption that the target density belongs to a Besov space (see [36]) makes reference to the effect of shifts on the distribution, as does shift-invariance. We do not see any obvious containments between classes of functions defined through shift-invariance and Besov spaces, but this is a potential topic for further research.

Another difference with prior work is the ability of our approach to succeed in the challenging noise-tolerant learning model. We are not aware of analyses for density estimation of densities belonging to Sobolev or Besov spaces that extend to the noise-tolerant setting in which the target density is only assumed to be close to some density in the relevant class.

As mentioned above, shift-invariance was used in the analysis of algorithms for learning discrete probability distributions in [5, 16]. Likewise, both the discrete and continuous Fourier transforms have been used in the past to learn discrete probability distributions [26, 27, 15].

## 2 Preliminaries

We write  $B(r)$  to denote the radius- $r$  ball in  $\mathbb{R}^d$ , i.e.  $B(r) = \{x \in \mathbb{R}^d : x_1^2 + \dots + x_d^2 \leq r^2\}$ . If  $f$  is a probability density over  $\mathbb{R}^d$  and  $S \subset \mathbb{R}^d$  is a subset of its domain, we write  $f_S$  to denote the density of  $f$  conditioned on  $S$ .

## 2.1 Shift-invariance

Roughly speaking, the shift-invariance of a distribution measures how much it changes (in total variation distance) when it is subjected to a small translation. The notion of shift-invariance has typically been used for discrete distributions (especially in the context of proving discrete limit theorems, see e.g. [13] and many references therein). We give a natural continuous analogue of this notion below.

► **Definition 1.** Given a probability density  $f$  over  $\mathbb{R}^d$ , a unit vector  $v$ , and a positive real value  $\kappa$ , we say that the *shift-invariance of  $f$  in direction  $v$  at scale  $\kappa$* , denoted  $\text{SI}(f, v, \kappa)$ , is

$$\text{SI}(f, v, \kappa) \stackrel{\text{def}}{=} \frac{1}{\kappa} \cdot \sup_{\kappa' \in [0, \kappa]} \int_{\mathbb{R}^d} |f(x + \kappa'v) - f(x)| dx. \quad (1)$$

Intuitively, if  $\text{SI}(f, v, \kappa) = \beta$ , then for any direction (unit vector)  $v$  the variation distance between  $f$  and a shift of  $f$  by  $\kappa'$  in direction  $v$  is at most  $\kappa\beta$  for all  $0 \leq \kappa' \leq \kappa$ . The factor  $\frac{1}{\kappa}$  in the definition means that  $\text{SI}(f, v, \kappa)$  does not necessarily go to zero as  $\kappa$  gets small; the effect of shifting by  $\kappa$  is measured relative to  $\kappa$ .

Let  $\text{SI}(f, \kappa) \stackrel{\text{def}}{=} \sup\{\text{SI}(f, v, \kappa) : v \in \mathbb{R}^d, \|v\|_2 = 1\}$ . For any constant  $c$  we define the class of densities  $\mathcal{C}_{\text{SI}}(c, d)$  to consist of all  $d$ -dimensional densities  $f$  with the property that  $\text{SI}(f, \kappa) \leq c$  for all  $\kappa > 0$ .

We could obtain an equivalent definition if we removed the factor  $\frac{1}{\kappa}$  from the definition of  $\text{SI}(f, v, \kappa)$ , and required that  $\text{SI}(f, v, \kappa) \leq c\kappa$  for all  $\kappa > 0$ . This could of course be generalized to enforce bounds on the modified  $\text{SI}(f, v, \kappa)$  that are not linear in  $\kappa$ . We have chosen to focus on linear bounds in this paper to have cleaner theorems and proofs.

We include “sup” in the definition due to the fact that smaller shifts can sometimes have bigger effects. For example, a sinusoid with period  $\xi$  is unaffected by a shift of size  $\xi$ , but profoundly affected by a shift of size  $\xi/2$ . Because of possibilities like this, to capture the intuitive notion that “small shifts do not lead to large changes”, we seem to need to evaluate the worst case over shifts of at most a certain size.

As described earlier, given a nonincreasing “tail bound” function  $g : \mathbb{R}^+ \rightarrow (0, 1)$  which is absolutely continuous and satisfies  $\lim_{t \rightarrow +\infty} g(t) = 0$ , we further define the class of densities  $\mathcal{C}_{\text{SI}}(c, d, g)$  to consist of those  $f \in \mathcal{C}_{\text{SI}}(c, d)$  which have the additional property that  $f$  has  *$g$ -light tails*, meaning that for all  $t > 0$ , it holds that  $\mathbf{Pr}_{\mathbf{x} \leftarrow f} [\|\mathbf{x} - \mu\| > t] \leq g(t)$ , where  $\mu \in \mathbb{R}^d$  is the mean of  $f$ .

► **Remark.** It will be convenient in our analysis to consider only tail bound functions  $g$  that satisfy  $\min\{r \in \mathbb{R} : g(r) \leq 1/2\} \geq 1/10$  (the constants  $1/2$  and  $1/10$  are arbitrary here and could be replaced by any other absolute positive constants). This is without loss of generality, since any tail bound function  $g$  which does not meet this criterion can simply be replaced by a weaker tail bound function  $g^*$  which does meet this criterion, and clearly if  $f$  has  $g$ -light tails then  $f$  also has  $g^*$ -light tails.

We will (ab)use the notation  $g^{-1}(\varepsilon)$  to mean  $\inf\{t : g(t) \leq \varepsilon\}$ .

The complexity of learning with a tail bound  $g$  will be expressed in part using  $I_g \stackrel{\text{def}}{=} \int_0^\infty g(\sqrt{z}) dz$ . We remark that the quantity  $I_g$  is the “right” quantity in the sense that the integral  $I_g$  is finite as long as the density has “non-trivial decay”. More precisely, note that by Chebyshev’s inequality,  $g(\sqrt{z}) = O(z^{-1})$ . Since the integral  $\int O(z^{-1}) dz$  diverges, this means that if  $I_g$  is finite, then the density  $f$  has a decay sharper than the trivial decay implied by Chebyshev’s inequality.

## 2.2 Fourier transform of high-dimensional distributions

In this subsection we gather some helpful facts from multidimensional Fourier analysis.

While it is possible to do Fourier analysis over  $\mathbb{R}^d$ , in this paper, we will only do Fourier analysis for functions  $f \in L_1([-1, 1]^d)$ .

► **Definition 2.** For any function  $f \in L_1([-1, 1]^d)$ , we define  $\widehat{f} : \mathbb{R}^d \rightarrow \mathbb{C}$  by  $\widehat{f}(\xi) = \int_{x \in \mathbb{R}^d} f(x) \cdot e^{\pi i \cdot \langle \xi, x \rangle} dx$ .

Next, we recall the following standard claims about Fourier transforms of functions, which may be found, for example, in [41].

► **Claim 3.** For  $f, g \in L_1([-1, 1]^d)$  let  $h(x) = \int_{y \in \mathbb{R}^d} f(y) \cdot g(x - y) dy$  denote the convolution  $h = f * g$  of  $f$  and  $g$ . Then for any  $\xi \in \mathbb{R}^d$ , we have  $\widehat{h}(\xi) = \widehat{f}(\xi) \cdot \widehat{g}(\xi)$ .

Next, we recall Parseval's identity on the cube.

► **Claim 4 (Parseval's identity).** For  $f : [-1, 1]^d \rightarrow \mathbb{R}$  such that  $f \in L_2([-1, 1]^d)$ , it holds that  $\int_{[-1, 1]^d} f(x)^2 dx = \frac{1}{2^d} \cdot \sum_{\xi \in \mathbb{Z}^d} |\widehat{f}(\xi)|^2$ .

The next claim says that the Fourier inversion formula can be applied to any sequence in  $\ell_2(\mathbb{Z}^d)$  to obtain a function whose Fourier series is identical to the given sequence.

► **Claim 5 (Fourier inversion formula).** For any  $g : \mathbb{Z}^d \rightarrow \mathbb{C}$  such that  $\sum_{\xi \in \mathbb{Z}^d} |g(\xi)|^2 < \infty$ , the function  $h(x) = \sum_{\xi \in \mathbb{Z}^d} \frac{1}{2^d} \cdot g(\xi) \cdot e^{\pi i \cdot \langle \xi, x \rangle}$ , is well defined and satisfies  $\widehat{h}(\xi) = g(\xi)$  for all  $\xi \in \mathbb{Z}^d$ .

We will also use Young's inequality:

► **Claim 6 (Young's inequality).** Let  $f \in L_p([-1, 1]^d)$ ,  $g \in L_q([-1, 1]^d)$ ,  $1 \leq p, q, r \leq \infty$ , such that  $1 + 1/r = 1/p + 1/q$ . Then  $\|f * g\|_r \leq \|f\|_p \cdot \|g\|_q$ .

## 2.3 A useful mollifier

Our algorithm and its analysis require the existence of a compactly supported distribution with fast decaying Fourier transform. Since the precise rate of decay is not very important, we use the  $C^\infty$  function  $b : [-1, 1] \rightarrow \mathbb{R}^+$  as follows:

$$b(x) = \begin{cases} c_0 \cdot e^{-\frac{x^2}{1-x^2}} & \text{if } |x| < 1 \\ 0 & \text{if } |x| = 1. \end{cases} \quad (2)$$

Here  $c_0 \approx 1.067$  is chosen so that  $b$  is a pdf; by symmetry, its mean is 0. (This function has previously been used as a mollifier [34, 28].) The following fact can be found in [32] (while it is proved only for  $\xi \in \mathbb{Z}$ , it is easy to see that the same proof holds if  $\xi \in \mathbb{R}$ ).

► **Fact 7.** For  $b : [-1, 1] \rightarrow \mathbb{R}^+$  defined in (2) and  $\xi \in \mathbb{Z} \setminus \{0\}$ , we have that  $|\widehat{b}(\xi)| \leq e^{-\sqrt{|\xi|}} \cdot |\xi|^{-3/4}$ .

Let us now define the function  $b_{d,\gamma} : \mathbb{R}^d \rightarrow \mathbb{R}^+$  as  $b_{d,\gamma}(x_1, \dots, x_d) = \frac{1}{\gamma^d} \cdot \prod_{j=1}^d b(x_j/\gamma)$ . Combining this definition and Fact 7, we have the following claim:

► **Claim 8.** For  $\xi \in \mathbb{Z}^d$  with  $\|\xi\|_\infty \geq t$ , we have  $|\widehat{b_{d,\gamma}}(\xi)| \leq e^{-\sqrt{\gamma \cdot t}} \cdot (\gamma \cdot t)^{-3/4}$ .

The next fact is immediate from (2) and the definition of  $b_{d,\gamma}$ :

► **Fact 9.**  $\|b_{d,\gamma}\|_\infty = (c_0/\gamma)^d$  and as a consequence,  $\|b_{d,\gamma}\|_2^2 \leq (c_0/\gamma)^{2d}$ .

### 3 A restricted problem: learning shift-invariant distributions with bounded support

As sketched in Section 1.2, we begin by presenting and analyzing a density estimation algorithm for densities that, in addition to being shift-invariant, have support bounded in  $B(1/2)$ . Our analysis also captures the fact that, to achieve accuracy  $\varepsilon$ , an algorithm often only needs the density to be learned to have shift invariance at a scale slightly finer than  $\varepsilon$ .

► **Lemma 10.** *There is an algorithm learn-bounded with the following property: For all constant  $d$ , for all  $\varepsilon, \delta > 0$ , all  $0 < \kappa < \varepsilon < 1/2$ , and all  $d$ -dimensional densities  $f$  with support in  $B(1/2)$  such that  $\kappa\text{SI}(f, \kappa) \leq \varepsilon/2$ , given access to independent draws from  $f$ , the algorithm runs in  $O_d\left(\frac{1}{\varepsilon^2} \left(\frac{1}{\kappa}\right)^{2d} \log^{4d}\left(\frac{1}{\kappa}\right) \log\left(\frac{1}{\kappa\delta}\right)\right)$  time uses  $O_d\left(\frac{1}{\varepsilon^2} \left(\frac{1}{\kappa}\right)^d \log^{2d}\left(\frac{1}{\kappa}\right) \log\left(\frac{1}{\kappa\delta}\right)\right)$  samples, and with probability  $1 - \delta$ , outputs a hypothesis  $h : [-1, 1]^d \rightarrow \mathbb{R}^+$  such that  $\int_{x \in \mathbb{R}^d} |f(x) - h(x)| \leq \varepsilon$ .*

*Further, given any point  $z \in [-1, 1]^d$ ,  $h(z)$  can be computed in time  $O_d\left(\frac{\log^{2d}(1/\kappa)}{\kappa^d}\right)$  and satisfies  $h(z) \leq O_d\left(\frac{\log^{2d}(1/\kappa)}{\kappa^d}\right)$ .*

**Proof.** Let  $0 < \gamma := \frac{\kappa}{\sqrt{d}}$ , and let us define  $q = f * b_{d,\gamma}$ . (Here  $*$  denotes convolution and  $b_{d,\gamma}$  is the mollifier defined in Section 2.3.) We make a few simple observations about  $q$ :

- (i) Since  $\gamma \leq 1/2$ , we have that  $q$  is a density supported on  $B(1)$ .
- (ii) Since  $d$  is a constant, a draw from  $b_{d,\gamma}$  can be generated in constant time. Thus given a draw from  $f$ , one can generate a draw from  $q$  in constant time, simply by generating a draw from  $b_{d,\gamma}$  and adding it to the draw from  $f$ .
- (iii) By Young's inequality (Claim 6), we have that  $\|q\|_2 \leq \|f\|_1 \cdot \|b_{d,\gamma}\|_2$ . Noting that  $f$  is a density and thus  $\|f\|_1 = 1$  and applying Fact 9, we obtain that  $\|q\|_2$  is finite. As a consequence, the Fourier coefficients of  $q$  are well-defined.

**Preliminary analysis.** We first observe that because  $b_{d,\gamma}$  is supported on  $[-\gamma, \gamma]^d$ , the distribution  $q$  may be viewed as an average of different shifts of  $f$  where each shift is by a distance at most  $\gamma\sqrt{d} \leq \kappa$ . Fix any direction  $v$  and consider a shift of  $f$  in direction  $v$  by some distance at most  $\gamma\sqrt{d} \leq \kappa$ . Since  $\kappa\text{SI}(f, \kappa) \leq \varepsilon/2$ , we have that the variation distance between  $f$  and this shift in direction  $v$  is at most  $\varepsilon/2$ . Averaging over all such shifts, it follows that  $d_{\text{TV}}(q, f) \leq \varepsilon/2$ .

Next, we observe that by Claim 3, for any  $\xi \in \mathbb{Z}^d$ , we have  $\widehat{q}(\xi) = \widehat{f}(\xi) \cdot \widehat{b_{d,\gamma}}(\xi)$ . Since  $f$  is a pdf,  $|\widehat{f}(\xi)| \leq 1$ , and thus we have  $|\widehat{q}(\xi)| \leq |\widehat{b_{d,\gamma}}(\xi)|$ . Also, for any parameter  $k \in \mathbb{Z}^+$ , define  $C_k = \{\xi \in \mathbb{Z}^d : \|\xi\|_\infty = k\}$ . Let us fix another parameter  $T$  (to be determined later). Applying Claim 8, we obtain

$$\begin{aligned} \sum_{\xi: \|\xi\|_\infty > T} |\widehat{q}(\xi)|^2 &\leq \sum_{\xi: \|\xi\|_\infty > T} |\widehat{b_{d,\gamma}}(\xi)|^2 \leq \sum_{k > T} \sum_{\xi: \|\xi\|_\infty = k} |\widehat{b_{d,\gamma}}(\xi)|^2 \\ &\leq \sum_{k > T} |C_k| \cdot e^{-2\sqrt{\gamma \cdot k}} \cdot (\gamma \cdot k)^{-3/2} \leq \sum_{k > T} (2k+1)^d \cdot e^{-2\sqrt{\gamma \cdot k}} \cdot (\gamma \cdot k)^{-3/2}. \end{aligned}$$

An easy calculation shows that if  $T \geq \frac{4d^2}{\gamma} \cdot \ln^2\left(\frac{d}{\gamma}\right)$ , then  $\sum_{\xi: \|\xi\|_\infty > T} |\widehat{q}(\xi)|^2 \leq 2(2T+1)^d \cdot e^{-2\sqrt{\gamma \cdot T}} \cdot (\gamma \cdot T)^{-3/2}$ . If we now set  $T$  to be  $\frac{4d^2}{\gamma} \cdot \ln^2\left(\frac{d}{\gamma}\right) + \frac{1}{\gamma} \cdot \ln^2\left(\frac{8}{\varepsilon}\right)$ , then  $\sum_{\xi: \|\xi\|_\infty > T} |\widehat{q}(\xi)|^2 \leq \frac{\varepsilon^2}{8}$ .

**The algorithm.** We first observe that for any  $\xi \in \mathbb{Z}^d$ , the Fourier coefficient  $\widehat{q}(\xi)$  can be estimated to good accuracy using relatively few draws from  $q$  (and hence from  $f$ , recalling (ii) above). More precisely, as an easy consequence of the definition of the Fourier transform, we have:

► **Observation 11.** *For any  $\xi \in \mathbb{Z}^d$ , the Fourier coefficient  $\widehat{q}(\xi)$  can be estimated to within additive error of magnitude at most  $\eta$  with confidence  $1 - \beta$  using  $O(1/\eta^2 \cdot \log(1/\beta))$  draws from  $q$ .*

Let us define the set **Low** of low-degree Fourier coefficients as  $\text{Low} = \{\xi \in \mathbb{Z}^d : \|\xi\|_\infty \leq T\}$ . Thus,  $|\text{Low}| \leq (2T+1)^d$ . Thus, using  $S = O(\eta^{-2} \cdot \log(T/\delta))$  draws from  $f$ , by Observation 11, with probability  $1 - \delta$ , we can compute a set of values  $\{\widehat{u}(\xi)\}_{\xi \in \text{Low}}$  such that

$$\text{For all } \xi \in \text{Low}, |\widehat{u}(\xi) - \widehat{q}(\xi)| \leq \eta. \quad (3)$$

Recalling (ii), the sequence  $\{\widehat{u}(\xi)\}_{\xi \in \text{Low}}$  can be computed in  $O(|S| \cdot |\text{Low}|)$  time. Define  $\widehat{u}(\xi) = 0$  for  $\xi \in \mathbb{Z}^d \setminus \text{Low}$ . Combining (3) with this, we get

$$\begin{aligned} \sum_{\xi \in \mathbb{Z}^d} |\widehat{u}(\xi) - \widehat{q}(\xi)|^2 &\leq \sum_{\xi \in \text{Low}} |\widehat{u}(\xi) - \widehat{q}(\xi)|^2 + \sum_{\xi \notin \text{Low}} |\widehat{u}(\xi) - \widehat{q}(\xi)|^2 \\ &\leq \sum_{\xi \in \text{Low}} |\widehat{u}(\xi) - \widehat{q}(\xi)|^2 + \frac{\varepsilon^2}{8} |\text{Low}| \cdot \eta^2 + \frac{\varepsilon^2}{8} \leq (2T+1)^d \cdot \eta^2 + \frac{\varepsilon^2}{8}. \end{aligned}$$

Thus, setting  $\eta$  as  $\eta^2 = (2T+1)^{-d} \cdot \frac{\varepsilon^2}{8}$ , we get that  $\sum_{\xi \in \mathbb{Z}^d} |\widehat{u}(\xi) - \widehat{q}(\xi)|^2 \leq \frac{\varepsilon^2}{4}$ . Note that by definition  $\widehat{u} : \mathbb{Z}^d \rightarrow \mathbb{C}$  satisfies  $\sum_{\xi \in \mathbb{Z}^d} |\widehat{u}(\xi)|^2 < \infty$ . Thus, we can apply the Fourier inversion formula (Claim 5) to obtain a function  $u : [-1, 1]^d \rightarrow \mathbb{C}$  such that

$$\int_{[-1, 1]^d} |u(x) - q(x)|^2 dx = \frac{1}{2^d} \cdot \left( \sum_{\xi \in \mathbb{Z}^d} |\widehat{u}(\xi) - \widehat{q}(\xi)|^2 \right) \leq \frac{\varepsilon^2}{4 \cdot 2^d}, \quad (4)$$

where the first equality follows by Parseval's identity (Claim 4). By the Cauchy-Schwarz inequality,  $\int_{[-1, 1]^d} |u(x) - q(x)| dx \leq \sqrt{2^d} \cdot \sqrt{\int_{[-1, 1]^d} |u(x) - q(x)|^2 dx}$ . Plugging in (4), we obtain  $\int_{[-1, 1]^d} |u(x) - q(x)| dx \leq \frac{\varepsilon}{2}$ . Let us finally define  $h$  (our final hypothesis),  $h : [-1, 1]^d \rightarrow \mathbb{R}^+$ , as follows:  $h(x) = \max\{0, \text{Re}(u(x))\}$ . Note that since  $q(x)$  is a non-negative real value for all  $x$ , we have

$$\int_{[-1, 1]^d} |h(x) - q(x)| dx \leq \int_{[-1, 1]^d} |u(x) - q(x)| dx \leq \frac{\varepsilon}{2}. \quad (5)$$

Finally, recalling that we previously proved  $d_{\text{TV}}(f, q) \leq \frac{\varepsilon}{2}$ , it follows that  $\int_{[-1, 1]^d} |h(x) - f(x)| dx \leq \varepsilon$ .

**Complexity analysis.** We now analyze the time and sample complexity of this algorithm as well as the complexity of computing  $h$ . First of all, observe that plugging in the value of  $\gamma$  and recalling that  $d$  is a constant, we get that  $T = \frac{4d^2}{\gamma} \cdot \ln^2\left(\frac{d}{\gamma}\right) + \frac{1}{\gamma} \cdot \ln^2\left(\frac{8}{\varepsilon}\right) = O\left(\frac{\log^2(1/\kappa)}{\kappa}\right)$ . Combining this with the choice of  $\eta$  we get that the algorithm uses

$$\begin{aligned} S &= O\left(\frac{1}{\eta^2} \cdot \log\left(\frac{|\text{Low}|}{\delta}\right)\right) = O\left(\frac{1}{\eta^2} \cdot \log\left(\frac{T}{\delta}\right)\right) = O\left(\frac{(2T+1)^d \cdot \log\left(\frac{T}{\delta}\right)}{\varepsilon^2}\right) \\ &= O_d\left(\frac{1}{\varepsilon^2} \left(\frac{1}{\kappa}\right)^d \log^{2d}\left(\frac{1}{\kappa}\right) \log\left(\frac{1}{\kappa\delta}\right)\right) \end{aligned}$$

draws from  $p$ . Next, as we have noted before, computing the sequence  $\{\widehat{u}(\xi)\}$  takes time

$$\begin{aligned} O(S \cdot |\text{Low}|) &= O_d\left(\frac{1}{\varepsilon^2} \left(\frac{1}{\kappa}\right)^d \log^{2d}\left(\frac{1}{\kappa}\right) \log\left(\frac{1}{\kappa\delta}\right) T^d\right) \\ &= O_d\left(\frac{1}{\varepsilon^2} \left(\frac{1}{\kappa}\right)^{2d} \log^{4d}\left(\frac{1}{\kappa}\right) \log\left(\frac{1}{\kappa\delta}\right)\right). \end{aligned}$$

To compute the function  $u$  (and hence  $h$ ) at any point  $x \in [-1, 1]^d$  takes time  $O(|\text{Low}|) = O_d\left(\frac{\log^{2d}(1/\kappa)}{\kappa^d}\right)$ . This is because the Fourier inversion formula (Claim 5) has at most  $O(|\text{Low}|)$  non-zero terms.

Finally, we prove the upper bound on  $h$ . If the training examples are  $x_1, \dots, x_S$ , then for any  $z \in [-1, 1]^d$ , we have

$$\begin{aligned} h(z) &\leq |u(z)| = \left| \sum_{\xi \in \text{Low}} \frac{1}{2^d} \cdot \widehat{u}(\xi) \cdot e^{\pi i \cdot \langle \xi, z \rangle} \right| = \left| \sum_{\xi \in \text{Low}} \frac{1}{2^d} \cdot \left( \frac{1}{S} \sum_{t=1}^S e^{\pi i \cdot \langle \xi, x_t \rangle} \right) \cdot e^{\pi i \cdot \langle \xi, z \rangle} \right| \\ &\leq \frac{|\text{Low}|}{2^d} = O_d\left(\frac{\log^{2d}(1/\kappa)}{\kappa^d}\right), \end{aligned}$$

completing the proof.  $\blacktriangleleft$

#### 4 Density estimation for densities in $\mathcal{C}_{\text{SI}}(c, d, g)$

Fix any nonincreasing tail bound function  $g : \mathbb{R}^+ \rightarrow [0, 1]$  which satisfies  $\lim_{t \rightarrow +\infty} g(t) = 0$  and  $\min\{r \in \mathbb{R} : g(r) \leq 1/2\} \geq 1/10$  and any constant  $c \geq 1$ . In this section we prove the following theorem which gives a density estimation algorithm for the class of distributions  $\mathcal{C}_{\text{SI}}(c, d, g)$ :

► **Theorem 12.** *For any  $c, g$  as above and any  $d \geq 1$ , there is an algorithm with the following property: Let  $f$  be any target density (unknown to the algorithm) which belongs to  $\mathcal{C}_{\text{SI}}(c, d, g)$ . Given any error parameter  $0 < \varepsilon < 1/2$  and confidence parameter  $\delta > 0$  and access to independent draws from  $f$ , the algorithm with probability  $1 - O(\delta)$  outputs a hypothesis  $h : [-1, 1]^d \rightarrow \mathbb{R}^{\geq 0}$  such that  $\int_{x \in \mathbb{R}^d} |f(x) - h(x)| \leq O(\varepsilon)$ .*

*The algorithm runs in  $O_{c,d}\left(\left((g^{-1}(\varepsilon))^{2d} \left(\frac{1}{\varepsilon}\right)^{2d+2} \log^{4d}\left(\frac{g^{-1}(\varepsilon)}{\varepsilon}\right) \log\left(\frac{g^{-1}(\varepsilon)}{\varepsilon\delta}\right) + I_g\right) \log \frac{1}{\delta}\right)$  time and uses  $O_{c,d}\left(\left((g^{-1}(\varepsilon))^d \left(\frac{1}{\varepsilon}\right)^{d+2} \log^{2d}\left(\frac{g^{-1}(\varepsilon)}{\varepsilon}\right) \log\left(\frac{g^{-1}(\varepsilon)}{\varepsilon\delta}\right) + I_g\right) \log \frac{1}{\delta}\right)$  samples.*

##### 4.1 Outline of the proof

Theorem 12 is proved by a reduction to Lemma 10. The main ingredient in the proof of Theorem 12 is a “transformation algorithm” with the following property: given as input access to i.i.d. draws from any density  $f \in \mathcal{C}_{\text{SI}}(c, d, g)$ , the algorithm constructs parameters which enable draws from the density  $f$  to be transformed into draws from another density, which we denote  $r$ . The density  $r$  is obtained by approximating  $f$  after conditioning on a non-tail sample, and scaling the result so that it lies in a ball of radius  $1/2$ .

Given such a transformation algorithm, the approach to learn  $f$  is clear: we first run the transformation algorithm to get access to draws from the transformed distribution  $r$ . We then use draws from  $r$  to run the algorithm of Lemma 10 to learn  $r$  to high accuracy. (Intuitively, the error relative to  $f$  of the final hypothesis density is  $O(\varepsilon)$  because at most



$O(\varepsilon)$  comes from the conditioning and at most  $O(\varepsilon)$  from the algorithm of Lemma 10.) We note that while this high-level approach is conceptually straightforward, a number of technical complications arise; for example, our transformation algorithm only succeeds with some non-negligible probability, so we must run the above-described combined procedure multiple times and perform hypothesis testing to identify a successful final hypothesis from the resulting pool of candidates.

The rest of this section is organized as follows: In Section 4.2 we give various necessary technical ingredients for our transformation algorithm. We state and prove the key results about the transformation algorithm in Section 4.3, and we use the transformation algorithm to prove Theorem 12 in Section 4.4.

## 4.2 Technical ingredients for the transformation algorithm

As sketched earlier, our approach will work with a density obtained by conditioning  $f \in \text{SI}(c, d)$  on lying in a certain ball that has mass close to 1 under  $f$ . While we know that the original density  $f \in \text{SI}(c, d)$  has good shift-invariance, we will further need the conditioned distribution to also have good shift-invariance in order for the learn-bounded algorithm of Section 3 to work. Thus we require the following simple lemma, which shows that conditioning a density  $f \in \text{SI}(c, d)$  on a region of large probability cannot hurt its shift invariance too much.

► **Lemma 13.** *Let  $f \in \text{SI}(c, d)$  and let  $B$  be a ball such that  $\Pr_{\mathbf{x} \sim f}[\mathbf{x} \in B] \geq 1 - \delta$  where  $\delta < 1/2$ . If  $f_B$  is the density of  $f$  conditioned on  $B$ , then, for all  $\kappa > 0$ ,  $\text{SI}(f_B, \kappa) \leq \frac{4\delta}{\kappa} + 2c$ .*

**Proof.** Let  $v$  be any unit vector in  $\mathbb{R}^d$ . Note that  $f$  can be expressed as  $(1 - \delta)f_B + \delta \cdot f_{err}$  where  $f_{err}$  is some other density. As a consequence, for any  $\kappa > 0$ , using the triangle inequality we have that

$$\int_{\mathbf{x}} |f(\mathbf{x}) - f(\mathbf{x} + \kappa v)| d\mathbf{x} \geq (1 - \delta) \int_{\mathbf{x}} |f_B(\mathbf{x}) - f_B(\mathbf{x} + \kappa v)| d\mathbf{x} - \delta \int_{\mathbf{x}} |f_{err}(\mathbf{x}) - f_{err}(\mathbf{x} + \kappa v)| d\mathbf{x}.$$

Since  $f \in \mathcal{C}_{\text{SI}}(c, d)$  the left hand side is at most  $c\kappa$ , whereas the subtrahend on the right hand side is trivially at most  $2\delta$ . Thus, we get  $\int_{\mathbf{x}} |f_B(\mathbf{x}) - f_B(\mathbf{x} + \kappa v)| d\mathbf{x} \leq \frac{2\delta}{1-\delta} + \frac{c\kappa}{1-\delta}$ , completing the proof. ◀

If  $f$  is an unknown target density then of course its mean is also unknown, and thus we will need to approximate it using draws from  $f$ . To do this, it will be helpful to convert our condition on the tails of  $f$  to bound the variance of  $\|\mathbf{x} - \mu\|$ , where  $\mathbf{x} \sim f$ .

► **Lemma 14.** *For any  $f \in \mathcal{C}_{\text{SI}}(c, d, g)$ , we have  $\mathbf{E}_{\mathbf{x} \sim f}[\|\mathbf{x} - \mu\|^2] \leq I_g$ .*

**Proof.** We have  $\mathbf{E}_{\mathbf{x} \sim f}[\|\mathbf{x} - \mu\|^2] = \int_0^\infty \Pr_{\mathbf{x} \sim f}[\|\mathbf{x} - \mu\|^2 \geq z] dz \leq \int_0^\infty g(\sqrt{z}) dz = I_g$ . ◀

The following easy proposition gives a guarantee on the quality of the empirical mean:

► **Lemma 15.** *For any  $f \in \mathcal{C}_{\text{SI}}(c, d, g)$ , if  $\mu \in \mathbb{R}^d$  is the mean of  $f$  and  $\hat{\mu}$  is its empirical estimate based on  $M$  samples, then for any  $t > 0$  we have  $\Pr[\|\mu - \hat{\mu}\|^2 \geq t] \leq \frac{I_g}{Mt}$ .*

**Proof.** If  $\mathbf{x}_1, \dots, \mathbf{x}_M$  are independent draws from  $f$ , then

$$\mathbf{E}[\|\mu - \hat{\mu}\|^2] = \mathbf{E}\left[\left\|\mu - \frac{\mathbf{x}_1 + \dots + \mathbf{x}_M}{M}\right\|^2\right] = \sum_{i=1}^M \frac{1}{M^2} \mathbf{E}\left[\|\mu - \mathbf{x}_i\|^2\right] = \frac{I_g}{M},$$

where the last inequality is by Lemma 14. Applying Markov's inequality on the left hand side, we get the stated claim. ◀

### 4.3 Transformation algorithm

► **Lemma 16.** *There is an algorithm compute-transformation such that given access to samples from  $f \in \mathcal{C}_{\text{SI}}(c, d, g)$  and an error parameter  $0 < \varepsilon < 1/2$ , the algorithm takes  $O(I_g)$  samples from  $f$  and with probability at least  $9/10$  produces a vector  $\tilde{\mu} \in \mathbb{R}^d$  and a real number  $t$  with the following properties: (1) For  $B_t = \{x : \|x - \tilde{\mu}\| \leq \sqrt{t}\}$ , we have  $\Pr_{\mathbf{x} \sim f}[\mathbf{x} \in B_t] \geq 1 - \varepsilon$ ; (2)  $t = O(g^{-1}(\varepsilon)^2)$ ; (3) For all  $\kappa > 0$ , the density  $f_{B_t}$  satisfies  $\text{SI}(f_{B_t}, \kappa) \leq \frac{4\varepsilon}{\kappa} + 2c$ .*

**Proof.** For  $M = 100I_g$ , the algorithm compute-transformation simply works as follows: set  $\tilde{\mu}$  to be the empirical mean of the  $M$  samples, and  $t = 2((g^{-1}(\varepsilon))^2 + 1/10)$ . (Note that since  $\min\{r \in \mathbb{R} : g(r) \leq 1/2\} \geq 1/10$  we have  $t = \Theta(g^{-1}(\varepsilon)^2)$ .) Let  $\mu$  denote the true mean of  $f$ . First, by Lemma 15, with probability at least  $0.9$ , the empirical mean  $\hat{\mu}$  will satisfy  $\|\mu - \hat{\mu}\|^2 \leq \frac{1}{10}$ . Let us assume for the rest of the proof that this happens; fix any such outcome and denote it  $\tilde{\mu}$ .

We have  $\|x - \tilde{\mu}\|^2 \leq 2(\|x - \mu\|^2 + \|\mu - \tilde{\mu}\|^2) \leq 2(\|x - \mu\|^2 + 1/10)$  and so

$$\Pr_{\mathbf{x} \in f}[\|\mathbf{x} - \tilde{\mu}\|^2 > t] \leq \Pr_{\mathbf{x} \in f}[2(\|\mathbf{x} - \mu\|^2 + 1/10) > t] = \Pr[\|\mathbf{x} - \mu\|^2 \geq g^{-1}(\varepsilon)] \leq \varepsilon.$$

Applying Lemma 13 completes the proof. ◀

The following proposition elaborates on the properties of the output of the transformation algorithm.

► **Lemma 17.** *Let  $f \in \mathcal{C}_{\text{SI}}(c, d, g)$ ,  $\varepsilon > 0$ ,  $\tilde{\mu} \in \mathbb{R}^d$ , and  $t \in \mathbb{R}$  satisfy the properties stated in Lemma 16. Consider the density  $f_{\text{scaled}}$  defined by  $f_{\text{scaled}}(x) \stackrel{\text{def}}{=} 2\sqrt{t} \cdot f(2\sqrt{t} \cdot (x + \tilde{\mu}))$  and  $f_{\text{scaled}}(x) \stackrel{\text{def}}{=} f_{\text{scaled}, B(1/2)}(x)$  where  $f_{\text{scaled}, B(1/2)}$  is the result of conditioning  $f_{\text{scaled}}$  on membership in  $B(1/2)$ . Then the density  $f_{\text{scaled}}$  satisfies the following properties: (1) The density  $f_{\text{scaled}}$  is supported in the ball  $B(1/2)$ ; (2) For all  $\varepsilon < 1/2$  and  $\kappa > 0$ , the density  $f_{\text{scaled}}$  satisfies  $\text{SI}(f_{\text{scaled}}, \kappa) \leq \frac{4\varepsilon}{\kappa} + 4c\sqrt{t}$ .*

**Proof.** First, it is easy to verify that function  $f_{\text{scaled}}$  defined above is indeed a density. Item 1 is enforced by fiat. Now, for any direction  $v$ , we have

$$\begin{aligned} \text{SI}(f_{\text{scaled}}, v, \kappa) &= \frac{1}{\kappa} \cdot \sup_{\kappa' \in [0, \kappa]} \int_{\mathbb{R}^d} |f_{\text{scaled}}(x + \kappa'v) - f_{\text{scaled}}(x)| dx \\ &= \frac{2\sqrt{t}}{\kappa} \cdot \sup_{\kappa' \in [0, \kappa]} \int_{\mathbb{R}^d} |f(2\sqrt{t}(x + \kappa'v)) - f(2\sqrt{t}x)| dx. \end{aligned}$$

Using a change of variables,  $u = 2\sqrt{t}x$ , we get

$$\begin{aligned} \text{SI}(f_{\text{scaled}}, v, \kappa) &= \frac{1}{\kappa} \cdot \sup_{\kappa' \in [0, \kappa]} \int_{\mathbb{R}^d} |f(u + \kappa'2\sqrt{t}v) - f(u)| du \\ &= \frac{1}{\kappa} \cdot \sup_{\kappa' \in [0, 2\sqrt{t}\kappa]} \int_{\mathbb{R}^d} |f(u + \kappa'v) - f(u)| du \\ &= 2\sqrt{t} \cdot \text{SI}(f, v, 2\sqrt{t}\kappa) \leq 2c\sqrt{t}. \end{aligned} \tag{6}$$

The last inequality uses that  $f \in \mathcal{C}_{\text{SI}}(c, d, g)$ . Inequality (6) implies that  $f_{\text{scaled}} \in \mathcal{C}_{\text{SI}}(2c\sqrt{t}, d, g)$ . Now,  $\Pr_{\mathbf{x} \sim f_{\text{scaled}}}(\mathbf{x} \in B(1/2)) = \Pr_{\mathbf{x} \sim f}(\mathbf{x} \in B_t) \geq 1 - \varepsilon$ , so applying Lemma 13 completes the proof. ◀

#### 4.4 Proof of Theorem 12

We are now ready to prove Theorem 12. Consider the following algorithm, which we call **construct-candidates**:

1. Run the transformation algorithm **compute-transformation**  $D := O(\ln(1/\delta))$  many times (with parameter  $\varepsilon$  each time). Let  $(\tilde{\mu}^{(i)}, t)$  be the output that it produces on the  $i$ -th run, where  $t = O(g^{-1}(\varepsilon)^2)$ .
2. For each  $i \in [D]$ , let  $B_t^{(i)} = \{x : \|x - \tilde{\mu}\| \leq \sqrt{t}\}$  and  $f_{\text{scnd}}^{(i)}$  be the density defined from  $(\tilde{\mu}^{(i)}, t)$  as in Lemma 17.

Before describing the third step of the algorithm, we observe that given the pair  $(\tilde{\mu}^{(i)}, t)$  it is easy to check whether any given  $x \in \mathbb{R}^d$  belongs to  $B_t^{(i)}$ . If  $\Pr_{\mathbf{x} \sim f}[\mathbf{x} \in B_t^{(i)}] \geq 1/2$ , then with probability at least  $1/2$  a draw from  $f$  can be used as a draw from  $f_{B_t^{(i)}}$ . In this case, via rejection sampling, it is easy to very efficiently simulate draws from  $f_{\text{scnd}}^{(i)}$  given access to samples from  $f$  (the average slowdown is at most a factor of 2). Note that if  $(\tilde{\mu}^{(i)}, t)$  satisfies the properties of Lemma 16, then  $\Pr_{\mathbf{x} \sim f}[\mathbf{x} \in B_t^{(i)}] \geq 1 - \varepsilon$  and we fall into this case. On the other hand, if  $\Pr_{\mathbf{x} \sim f}[\mathbf{x} \in B_t^{(i)}] < 1/2$ , then it may be inefficient to simulate draws from  $f_{\text{scnd}}^{(i)}$ . But any such  $i$  will not satisfy the properties of Lemma 16, so if rejection sampling is inefficient to simulate draws from  $f_{\text{scnd}}^{(i)}$  then we can ignore such an  $i$  in what follows. With this in mind, the third and fourth steps of the algorithm are as follows:

3. For each  $i \in [D]$ ,<sup>4</sup> run the algorithm **learn-bounded** using  $m$  samples from  $f_{\text{scnd}}^{(i)}$ , where  $m = m(\varepsilon, \delta, d)$  is the sample complexity of **learn-bounded** from Lemma 10. Let  $h_{\text{scnd}}^{(i)}$  be the resulting hypothesis that **learn-bounded** outputs.
4. Finally, for each  $i \in [D]$  output the hypothesis obtained by inverting the mapping of Lemma 17, i.e.

$$h^{(i)}(x) \stackrel{\text{def}}{=} \frac{1}{2\sqrt{t}} \cdot h_{\text{scnd}}^{(i)} \left( \frac{1}{2\sqrt{t}} \cdot (x - \tilde{\mu}^{(i)}) \right). \quad (7)$$

Thus the output of **construct-candidate** is a  $D$ -tuple of hypotheses  $(h^{(1)}, \dots, h^{(D)})$ .

We now analyze the **construct-candidate** algorithm. Given Lemma 16 and Lemma 17, it is not difficult to show that with high probability at least one of the hypotheses that it outputs has error  $O(\varepsilon)$  with respect to  $f$ :

► **Lemma 18.** *With probability at least  $1 - O(\delta)$ , at least one  $h^{(i)}$  has  $\int_x |h^{(i)}(x) - f(x)| dx \leq O(\varepsilon)$ .*

**Proof.** It is immediate from Lemma 16 and the choice of  $D$  that with probability  $1 - \delta$  at least one triple  $(\tilde{\mu}^{(i)}, t)$  satisfies the properties of Lemma 16. Fix  $i'$  to be an  $i$  for which this holds.

Given any  $i \in [D]$ , it is easy to carry out the check for whether rejection sampling is too inefficient in simulating  $f_{\text{scnd}}^{(i)}$  in such a way that algorithm **learn-bounded** will indeed be run to completion (as opposed to being terminated) on  $f_{\text{scnd}}^{(i')}$  with probability at least  $1 - \delta$ , so we henceforth suppose that indeed **learn-bounded** is actually run to completion on  $f_{\text{scnd}}^{(i')}$ . Since  $(\tilde{\mu}^{(i')}, t)$  satisfies the properties of Lemma 16, by Lemma 17, taking

<sup>4</sup> Actually, as described above, this and the fourth step are done only for those  $i$  for which rejection sampling is not too inefficient in simulating draws from  $f_{\text{scnd}}^{(i)}$  given draws from  $f$ ; for the other  $i$ 's, the run of **learn-bounded** is terminated.

$\kappa = \min\{\varepsilon/2, \varepsilon/(4g^{-1}(\varepsilon)c)\}$  the density  $f_{\text{scnd}}^{(i')}$  satisfies the required conditions for Lemma 10 to apply with that choice of  $\kappa$ . The following simple proposition, proved in the long version of this paper [20], implies that  $h^{(i)}$  is likewise  $O(\varepsilon)$ -close to  $f_{B_i}$ :

► **Proposition 19.** *Let  $f$  and  $g$  be two densities in  $\mathbb{R}^d$  and let  $x \mapsto A(x - z)$  be any invertible linear transformation over  $\mathbb{R}^d$ . Let  $f_A(x) = \det(A) \cdot f(A(x - z))$  and  $g_A(x) = \det(A) \cdot g(A(x - z))$  be the densities from  $f$  and  $g$  under this transformation. Then  $d_{\text{TV}}(f, g) = d_{\text{TV}}(f_A, g_A)$ .*

It remains only to observe that by property 1 of Lemma 16 the density  $f_{B_i}$  is  $\varepsilon$ -close to  $f$ , and then by the triangle inequality we have that  $h^{(i)}$  is  $O(\varepsilon)$ -close to  $f$ . This gives Lemma 18. ◀

Tracing through the parameters, it is straightforward to verify that the sample and time complexities of **construct-candidates** are as claimed in the statement of Theorem 12. These sample and time complexities dominate the sample and time complexities of the remaining portion of the algorithm, the hypothesis selection procedure discussed below.

All that is left is to identify a good hypothesis from the pool of  $D$  candidates. This can be carried out rather straightforwardly using well-known tools for hypothesis selection. Many variants of the basic hypothesis selection procedure have appeared in the literature, see e.g. [44, 18, 2, 17, 19]). The following is implicit in the proof of Proposition 6 from [19]:

► **Proposition 20.** *Let  $\mathbf{D}$  be a distribution with support contained in a set  $W$  and let  $\mathcal{D}_\varepsilon = \{\mathbf{D}_j\}_{j=1}^M$  be a collection of  $M$  hypothesis distributions over  $W$  with the property that there exists  $i \in [M]$  such that  $d_{\text{TV}}(\mathbf{D}, \mathbf{D}_i) \leq \varepsilon$ . There is an algorithm  $\text{Select}^{\mathbf{D}}$  which is given  $\varepsilon$  and a confidence parameter  $\delta$ , and is provided with access to (i) a source of i.i.d. draws from  $\mathbf{D}$  and from  $\mathbf{D}_i$ , for all  $i \in [M]$ ; and (ii) a  $(1 + \beta)$  “approximate evaluation oracle”  $\text{eval}_{\mathbf{D}_i}(\beta)$ , for each  $i \in [M]$ , which, on input  $w \in W$ , deterministically outputs  $\tilde{D}_i^\beta(w)$  such that the value  $\frac{\mathbf{D}_i(w)}{1+\beta} \leq \tilde{D}_i^\beta(w) \leq (1 + \beta) \cdot \mathbf{D}_i(w)$ . Further,  $(1 + \beta)^2 \leq (1 + \varepsilon/8)$ . The  $\text{Select}^{\mathbf{D}}$  algorithm has the following behavior: It makes  $m = O((1/\varepsilon^2) \cdot (\log M + \log(1/\delta)))$  draws from  $\mathbf{D}$  and from each  $\mathbf{D}_i$ ,  $i \in [M]$ , and  $O(m)$  calls to each oracle  $\text{eval}_{\mathbf{D}_i}$ ,  $i \in [M]$ . It runs in time  $\text{poly}(m, M)$  (counting each call to an  $\text{eval}_{\mathbf{D}_i}$  oracle and draw from a  $\mathbf{D}_i$  distribution as unit time), and with probability  $1 - \delta$  it outputs an index  $i^* \in [M]$  that satisfies  $d_{\text{TV}}(\mathbf{D}, \mathbf{D}_{i^*}) \leq 6\varepsilon$ .*

As suggested above, the remaining step is to apply Proposition 20 to the list of candidate hypothesis  $h^{(i)}$  which satisfies the guarantee of Lemma 18. However, to bound the sample and time complexity of running the procedure Proposition 20, we need to bound the complexity both of sampling from  $\{h^{(i)}\}_{i \in [D]}$  as well as of constructing approximate evaluation oracles for these measures.<sup>5</sup> In fact, we will first construct densities out of the measures  $\{h^{(i)}\}_{i \in [D]}$  and show how to both efficiently sample from these measures as well as construct approximate evaluation oracles for these densities.

Towards this, let us now define  $H_{\max}$  as follows:  $H_{\max} = \max_{i \in [D]} \max_{z \in [-1, 1]^n} h_{\text{scnd}}^{(i)}(z)$ . From Lemma 10 (recall that Lemma 10 was applied with  $\kappa = \min\{\varepsilon/2, \varepsilon/(4g^{-1}(\varepsilon)c)\}$ ) we get that  $H_{\max} = O_{c,d} \left( \left( \frac{g^{-1}(\varepsilon)}{\varepsilon} \right)^d \log^{2d} \frac{g^{-1}(\varepsilon)}{\varepsilon} \right)$ . We will carry out the rest of our calculations in terms of  $H_{\max}$ .

<sup>5</sup> Note that while  $h^{(i)}$  are forced to be non-negative and thus can be seen as measures, they need not integrate to 1 and thus need not be densities.

► **Observation 21.** For any  $i \in [D]$ ,  $\int_{x \in [-1,1]^d} h_{\text{scnd}}^{(i)}(x) dx$  can be estimated to additive accuracy  $\pm \varepsilon$  and confidence  $1 - \delta$  in time  $O_d \left( \frac{H_{\text{max}}^2}{\varepsilon^2} \cdot \log(1/\delta) \right)$ .

**Proof.** First note that it suffices to estimate the quantity  $\mathbf{E}_{x \in [-1,1]^d} [h_{\text{scnd}}^{(i)}(x)]$  to additive error  $\varepsilon/2^d$ . However, this can be estimated using the trivial random sampling algorithm. In particular, as  $h_{\text{scnd}}^{(i)}(x) \in [0, H_{\text{max}}]$ , the variance of the simple unbiased estimator for  $\mathbf{E}_{x \in [-1,1]^d} [h_{\text{scnd}}^{(i)}(x)]$  is also bounded by  $H_{\text{max}}^2$ . This finishes the proof. ◀

Note that, while the algorithm of Observation 21 does random sampling, this sampling is not from  $f$ , so it adds nothing to the sample complexity of the learning algorithm.

Next, for  $i \in [D]$ , let us define the quantity  $Z_i$  to be  $Z_i = \int_x h^{(i)}(x) dx$ . Since the functions  $h^{(i)}$  and  $h_{\text{scnd}}^{(i)}$  are obtained from each other by linear transformations (recall (7)), we get that  $2\sqrt{t}Z_i = \int_x h_{\text{scnd}}^{(i)} \left( \frac{1}{2\sqrt{t}} \cdot (x - \tilde{\mu}^{(i)}) \right) dx$ . We now define the functions  $H^{(i)}$  and  $H_{\text{scnd}}^{(i)}$  as  $H^{(i)}(x) = \frac{h^{(i)}(x)}{Z_i}$  and  $H_{\text{scnd}}^{(i)}(x) = \frac{h_{\text{scnd}}^{(i)} \left( \frac{1}{2\sqrt{t}} \cdot (x - \tilde{\mu}^{(i)}) \right)}{Z_i} \cdot \frac{1}{2\sqrt{t}}$ . Observe that the functions  $H^{(i)}$  and  $H_{\text{scnd}}^{(i)}$  are densities (i.e. they are non-negative and integrate to 1). First, we will show that it suffices to run the procedure  $\text{Select}^{\mathbf{D}}$  on the densities  $H^{(i)}$ . To see this, note that Lemma 18 says that there exists  $i \in [D]$  such that  $h^{(i)}$  satisfies  $\int_x |h^{(i)}(x) - f(x)| = O(\varepsilon)$ . For such an  $i$ ,  $Z_i \in [1 - O(\varepsilon), 1 + O(\varepsilon)]$ . Thus, we have the following corollary.

► **Corollary 22.** With probability at least  $1 - \delta$ , at least one  $H^{(i)}$  satisfies  $\int_x |H^{(i)}(x) - f(x)| = O(\varepsilon)$ . Further, for such an  $i$ ,  $Z_i \in [1 - O(\varepsilon), 1 + O(\varepsilon)]$ .

Thus, it suffices to run the procedure  $\text{Select}^{\mathbf{D}}$  on the candidate distributions  $\{H^{(i)}\}_{i \in [D]}$ . The next proposition shows that the densities  $\{H^{(i)}\}_{i \in [D]}$  are samplable.

► **Proposition 23.** A draw from the density  $H^{(i)}(x)$  can be sampled in time  $O(H_{\text{max}}/Z_i)$ .

**Proof.** First of all, note that it suffices to sample from  $H_{\text{scnd}}^{(i)}$  since  $H^{(i)}$  and  $H_{\text{scnd}}^{(i)}$  are linear transformations of each other. However, sampling from  $H_{\text{scnd}}^{(i)}$  is easy using rejection sampling. More precisely, the distribution  $H_{\text{scnd}}^{(i)}$  is supported on  $[-1, 1]^d$ . We sample from  $H_{\text{scnd}}^{(i)}$  as follows:

1. Let  $C = [-1, 1]^d \times [0, H_{\text{max}}]$ . Sample a uniformly random point  $z' = (z_1, \dots, z_{d+1})$  from  $C$ .
2. If  $z_{d+1} \leq h_{\text{scnd}}^{(i)}(z_1, \dots, z_d)$ , then return the point  $z = (z_1, \dots, z_d)$ .
3. Else go to Step 1 and repeat.

Now note that conditioned on returning a point in step 2, the point  $z$  is returned with probability proportional to  $h_{\text{scnd}}^{(i)}(z)$ . Thus, the distribution sampled by this procedure is indeed  $H_{\text{scnd}}^{(i)}(z)$ . To bound the probability of success, note that the total volume of  $C$  is  $2^d \times H_{\text{max}}$ . On the other hand, step 2 is successful only if  $z'$  falls in a region of volume  $Z_i$ . This finishes the proof. ◀

The next proposition says that if  $Z_i \geq 1/2$ , then there is an approximate evaluation oracle for the density  $H^{(i)}$ .

► **Proposition 24.** Suppose  $Z_i \geq 1/2$ . Then there is a  $(1 + O(\varepsilon))$ -approximate evaluation oracle for  $H^{(i)}$  which can be computed at any point  $w$  in time  $O \left( \frac{H_{\text{max}}^2}{\varepsilon^2} \right)$ .

**Proof.** Note that we can evaluate  $h^{(i)}$  at any point  $w$  exactly and thus the only issue is to estimate the normalizing factor  $Z_i$ . Note that since  $Z_i \geq 1/2$ , estimating  $Z_i$  to within an additive  $O(\varepsilon)$  gives us a  $(1 + O(\varepsilon))$  multiplicative approximation to  $Z_i$  and hence to  $H^{(i)}(w)$  at any point  $w$ . However, by Observation 21, this takes time  $O\left(\frac{H_{\max}^2}{\varepsilon^2}\right)$ , concluding the proof.  $\blacktriangleleft$

We now apply Proposition 20 as follows.

1. For all  $i \in [D]$ , estimate  $Z_i$  using Observation 21 up to an additive error  $\varepsilon$ . Let the estimates be  $\widehat{Z}_i$ .
2. Let us define  $S_{\text{feas}} = \{i \in [D] : \widehat{Z}_i \geq 1/2\}$ .
3. We run the routine **Select<sup>D</sup>** on the densities  $\{H^{(i)}\}_{i \in S_{\text{feas}}}$ . To sample from a density  $H^{(i)}$ , we use Proposition 23. We also construct a  $\beta = \varepsilon/32$  approximation oracle for each of the densities  $H^{(i)}$  using Proposition 24. Return the output of **Select<sup>D</sup>**.

The correctness of the procedure follows quite easily. Namely, note that Corollary 22 implies that there is one  $i$  such that both  $Z_i \in [1 - O(\varepsilon), 1 + O(\varepsilon)]$  and  $\int_x |H^{(i)}(x) - f(x)| = O(\varepsilon)$ . Thus such an  $i$  will be in  $S_{\text{feas}}$ . Thus, by the guarantee of **Select<sup>D</sup>**, the output hypothesis is  $O(\varepsilon)$  close to  $f$ .

We now bound the sample complexity and time complexity of this hypothesis selection portion of the algorithm. First of all, the number of samples required from  $f$  for running **Select<sup>D</sup>** is  $O((1/\varepsilon^2) \cdot (\log(1/\delta) + d^2 \log d + \log \log(1/\delta))) = O((1/\varepsilon^2) \cdot (\log(1/\delta) + d^2 \log d))$ . This is clearly dominated by the sample complexity of the previous parts. To bound the time complexity, note that the time complexity of invoking the sampling oracle for any  $H^{(i)}$  ( $i \in S_{\text{feas}}$ ) is dominated by the time complexity of the approximate oracle which is  $2^{O(d)} \cdot H_{\max}^2/\varepsilon^2$ . The total number of calls to the sampling as well as evaluation oracle is upper bounded by  $\frac{1}{\varepsilon^2}(D \log D + D \log(1/\delta))$ . Plugging in the value of  $H_{\max}$  as well as  $D$ , we see that the total time complexity is dominated by the bound in the statement of Theorem 12. This finishes the proof.

## 5 Efficiently learning multivariate log-concave densities

In this section we present our main application, which is an efficient algorithm for learning  $d$ -dimensional log-concave densities. We prove the following:

► **Theorem 25.** *There is an algorithm with the following property: Let  $f$  be a unknown log-concave density over  $\mathbb{R}^d$ . Given any error parameter  $\varepsilon > 0$  and confidence parameter  $\delta > 0$  and access to independent draws from  $f$ , the algorithm with probability  $1 - \delta$  outputs a hypothesis density  $h : \mathbb{R}^d \rightarrow \mathbb{R}^{\geq 0}$  such that  $\int_{x \in \mathbb{R}^d} |f(x) - h(x)| \leq O(\varepsilon)$ . The algorithm runs in time  $O_d\left(\left(\frac{1}{\varepsilon}\right)^{2d+2} \log^{7d}\left(\frac{1}{\varepsilon}\right) \log\left(\frac{1}{\varepsilon\delta}\right) \log\left(\frac{1}{\delta}\right)\right)$  and uses  $O_d\left(\left(\frac{1}{\varepsilon}\right)^{d+2} \log^{4d}\left(\frac{1}{\varepsilon}\right) \log\left(\frac{1}{\varepsilon\delta}\right) \log\left(\frac{1}{\delta}\right)\right)$  samples.*

We will establish Theorem 25 in two stages. First, we will show that any log-concave  $f$  that is nearly isotropic in fact belongs to a suitable class  $\mathcal{C}_{\text{SI}}(c, d)$ ; given this, the theorem follows immediately from Theorem 12 and a straightforward tracing through of the resulting time and sample complexity bounds. Then, we will reduce to the near-isotropic case, similarly to what was done in [37, 4].

First, let us state the theorem for the well-conditioned case. For this, the following definitions will be helpful.

► **Definition 26.** Let  $\Sigma$  and  $\tilde{\Sigma}$  be two positive semidefinite matrices. We say that  $\Sigma$  and  $\tilde{\Sigma}$  are  $C$ -approximations of each other (denoted by  $\Sigma \approx_C \tilde{\Sigma}$ ) if for every  $x \in \mathbb{R}^n$  such that  $x^T \tilde{\Sigma} x \neq 0$ , we have  $\frac{1}{C} \leq \frac{x^T \Sigma x}{x^T \tilde{\Sigma} x} \leq C$ .

► **Definition 27.** Say that the probability distribution is  $C$ -nearly-isotropic if its covariance matrix  $C$ -approximates  $I$ , the  $d$ -by- $d$  identity matrix.

► **Theorem 28.** *There is an algorithm with the following property: Let  $f$  be a unknown  $C$ -nearly-isotropic log-concave density over  $\mathbb{R}^d$ , where  $C$  and  $d$  are constants.*

*Given any error parameter  $\varepsilon > 0$  and confidence parameter  $\delta > 0$  and access to independent draws from  $f$ , the algorithm with probability  $1 - \delta$  outputs a hypothesis density  $h : \mathbb{R}^d \rightarrow \mathbb{R}^{\geq 0}$  such that  $\int_{x \in \mathbb{R}^d} |f(x) - h(x)| \leq O(\varepsilon)$ . The algorithm runs in time  $O_{C,d} \left( \left( \frac{1}{\varepsilon} \right)^{2d+2} \log^{7d} \left( \frac{1}{\varepsilon} \right) \log \left( \frac{1}{\varepsilon \delta} \right) \log \frac{1}{\delta} \right)$  and uses  $O_{C,d} \left( \left( \frac{1}{\varepsilon} \right)^{d+2} \log^{4d} \left( \frac{1}{\varepsilon} \right) \log \left( \frac{1}{\varepsilon \delta} \right) \log \frac{1}{\delta} \right)$  samples.*

By Theorem 12, Theorem 28 is an immediate consequence of the following theorem on the shift-invariance of near-isotropic log-concave distributions.

► **Theorem 29.** *Let  $f$  be a  $C$ -nearly-isotropic log-concave density in  $\mathbb{R}^d$ , for constants  $C$  and  $d$ . Then, for  $g(t) = e^{-\Omega(t)}$ , there is a constant  $c_1 = O_{C,d}(1)$  such that  $f \in \mathcal{C}_{\text{SI}}(c_1, d, g)$ .*

**Proof.** The fact that  $f$  has  $e^{-\Omega(t)}$ -light tails directly follows from Lemma 5.17 of [37], so it remains to prove that there is a constant  $c_1$  such that  $f \in \mathcal{C}_{\text{SI}}(c_1, d)$ . Because membership in  $\mathcal{C}_{\text{SI}}(c_1, d)$  requires that a condition be satisfied for all directions  $v$ , rotating a distribution does not affect its membership in  $\mathcal{C}_{\text{SI}}(c_1, d)$ .

Choose a unit vector  $v$  and  $\kappa > 0$ . By rotating the distribution if necessary, we may assume that  $v = e_1$ , and our goal of showing that  $\text{SI}(f, e_1, \kappa) \leq c_1$  is equivalent to showing that  $\int |f(x) - f(x + \kappa' e_1)| dx \leq c_1 \kappa$  for all  $\kappa' \leq \kappa$ .

We bound the integral of the LHS as follows. Fix some value of  $x' \stackrel{\text{def}}{=} (x_2, \dots, x_d)$ . Let us define  $L_{x'} \stackrel{\text{def}}{=} \{(x_1, x_2, \dots, x_d) : x_1 \in \mathbb{R}\}$  to be the line through  $(0, x_2, \dots, x_d)$  and  $(1, x_2, \dots, x_d)$ . Since the restriction of a concave function to a line is concave, the restriction of a log-concave distribution to a line is log-concave. Since

$$\int |f(x) - f(x + \kappa' e_1)| dx = \int_{x'} \int_{x_1} |f(x_1, x_2, \dots, x_d) - f(x_1 + \kappa', x_2, \dots, x_d)| dx_1 dx' \quad (8)$$

we are led to examine the one-dimensional log-concave measure  $f(\cdot, x_2, \dots, x_d)$ . The following will be useful for that.

► **Claim 30.** *Let  $\ell : \mathbb{R} \rightarrow \mathbb{R}$  be a log-concave measure. Then,  $\int |\ell(t) - \ell(t + h)| dt \leq 3h \cdot \max_{t \in \mathbb{R}} \ell(t)$ .*

**Proof.** Log-concave measures are unimodal (see [31]). Let  $z$  be the mode of  $\ell$ , so that  $\ell$  is non-decreasing on the interval  $[-\infty, z]$  and non-increasing in  $[z, \infty]$ . We have

$$\begin{aligned} & \int |\ell(t) - \ell(t + h)| dt \\ &= \int_{-\infty}^{z-h} |\ell(t) - \ell(t + h)| dt + \int_{z-h}^z |\ell(t) - \ell(t + h)| dt + \int_z^{\infty} |\ell(t) - \ell(t + h)| dt \end{aligned}$$



$$\begin{aligned}
 &= \int_{-\infty}^{z-h} \ell(t+h) - \ell(t) dt + \int_{z-h}^z |\ell(t) - \ell(t+h)| dt + \int_z^{\infty} \ell(t) - \ell(t+h) dt \\
 &\hspace{10em} \text{(since } z \text{ is the mode of } \ell) \\
 &= \int_{z-h}^z \ell(t) dt + \int_{z-h}^z |\ell(t) - \ell(t+h)| dt + \int_z^{z+h} \ell(t) dt \leq 3h \max_{t \in \mathbb{R}} \ell(t). \quad \blacktriangleleft
 \end{aligned}$$

Returning to the proof of Theorem 29, applying Claim 30 with (8), we get

$$\int |f(x) - f(x + \kappa' e_1)| dx \leq 3\kappa' \int_{x'} \left( \max_{x_1 \in L_{x'}} f(x_1, x') \right) dx'. \quad (9)$$

Now, since an isotropic log-concave distribution  $g$  satisfies  $g(x) \leq K \exp(-\|x\|)$  for an absolute constant  $K$  (see Theorem 5.1 of [40]), our  $C$ -nearly-isotropic log-concave distribution  $f$  satisfies  $f(x) \leq C^d K \exp(-\|x\|) = O_{C,d}(\exp(-\|x\|))$ . Plugging this into (9), we get

$$\begin{aligned}
 \int |f(x) - f(x + \kappa' e_1)| dx &\leq O_{C,d}(\kappa') \int_{x'} \left( \max_{x_1 \in L_{x'}} \exp(-\|(x_1, x')\|) \right) dx' \\
 &\leq O_{C,d}(\kappa') \int_{x'} \exp(-\|x'\|) dx'.
 \end{aligned}$$

Since the integral converges, this finishes the proof.  $\blacktriangleleft$

To learn log-concave distributions that are not  $C$ -nearly-isotropic, using techniques from [37], we preprocess the data to bring it into isotropic position, and then apply Theorem 29. The details are in the long version of this paper [20].

## 6 Learning shift-invariant densities over $\mathbb{R}^d$ with bounded support requires $\Omega(1/\varepsilon^d)$ samples

The following lower bound is proved in the long version of this paper [20].

**► Theorem 31.** *Given  $d \geq 1$ , there is a constant  $c_d = \Theta(\sqrt{d})$  such that the following holds: For all sufficiently small  $\varepsilon$ , let  $A$  be an algorithm with the following property: given access to  $m$  i.i.d. samples from an arbitrary (and unknown) finitely supported density  $f \in \mathcal{C}_{\text{SI}}(c_d, d)$ , with probability at least 99/100,  $A$  outputs a hypothesis density  $h$  such that  $d_{\text{TV}}(f, h) \leq \varepsilon$ . Then  $m \geq \Omega((1/\varepsilon)^d)$ .*

---

### References

- 1 J. Acharya, C. Daskalakis, and G. Kamath. Optimal Testing for Properties of Distributions. In *NIPS*, pages 3591–3599, 2015.
- 2 J. Acharya, A. Jafarpour, A. Orlitsky, and A.T. Suresh. Near-optimal-sample estimators for spherical Gaussian mixtures, 2014. <http://arxiv.org/abs/1402.4746>.
- 3 Jayadev Acharya, Ilias Diakonikolas, Jerry Li, and Ludwig Schmidt. Sample-optimal density estimation in nearly-linear time. In *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1278–1289. SIAM, 2017.
- 4 Maria-Florina Balcan and Philip M Long. Active and passive learning of linear separators under log-concave distributions. In *Conference on Learning Theory*, pages 288–316, 2013.
- 5 Andrew D Barbour and Aihua Xia. Poisson perturbations. *ESAIM: Probability and Statistics*, 3:131–150, 1999.
- 6 Andrew R Barron and Thomas M Cover. Minimum complexity density estimation. *IEEE transactions on information theory*, 37(4):1034–1054, 1991.

- 7 OV Besov. On a family of function spaces. Embedding and extension theorems. *Dokl. Akad. Nauk SSSR*, 126(6):1163–1165, 1959.
- 8 Aditya Bhaskara, Ananda Suresh, and Morteza Zadimoghaddam. Sparse solutions to non-negative linear systems and applications. In *Artificial Intelligence and Statistics*, pages 83–92, 2015.
- 9 C. L. Canonne, I. Diakonikolas, T. Gouleakis, and R. Rubinfeld. Testing Shape Restrictions of Discrete Distributions. In *STACS*, pages 25:1–25:14, 2016.
- 10 T. Carpenter, I. Diakonikolas, A. Sidiropoulos, and A. Stewart. Near-Optimal Sample Complexity Bounds for Maximum Likelihood Estimation of Multivariate Log-concave Densities. *CoRR*, abs/1802.10575, 2018.
- 11 S. Chan, I. Diakonikolas, R. Servedio, and X. Sun. Learning mixtures of structured distributions over discrete domains. In *SODA*, pages 1380–1394, 2013.
- 12 S. Chan, I. Diakonikolas, R. Servedio, and X. Sun. Efficient Density Estimation via Piecewise Polynomial Approximation. In *STOC*, pages 604–613, 2014.
- 13 L. Chen, L. Goldstein, and Q.-M. Shao. *Normal Approximation by Stein’s Method*. Springer, 2011.
- 14 S. Dasgupta. Learning mixtures of Gaussians. In *Proceedings of the 40th Annual Symposium on Foundations of Computer Science*, pages 634–644, 1999.
- 15 C. Daskalakis, A. De, G. Kamath, and C. Tzamos. A size-free CLT for Poisson multinomials and its applications. In *STOC*, pages 1074–1086, 2016.
- 16 C. Daskalakis, I. Diakonikolas, R. O’Donnell, R. A. Servedio, and L. Tan. Learning sums of independent integer random variables. In *FOCS*, pages 217–226, 2013.
- 17 C. Daskalakis, I. Diakonikolas, and R.A. Servedio. Learning Poisson Binomial Distributions. In *STOC*, pages 709–728, 2012.
- 18 C. Daskalakis and G. Kamath. Faster and Sample Near-Optimal Algorithms for Proper Learning Mixtures of Gaussians. In *COLT*, pages 1183–1213, 2014.
- 19 A. De, I. Diakonikolas, and R. Servedio. Learning from Satisfying Assignments. In *Proc. ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 478–497, 2015.
- 20 A. De, P. M. Long, and R. A. Servedio. Density estimation for shift-invariant multidimensional distributions, 2018. [arXiv:1811.03744](https://arxiv.org/abs/1811.03744).
- 21 A. De, P. M. Long, and R. A. Servedio. Learning Sums of Independent Random Variables with Sparse Collective Support. *FOCS*, 2018.
- 22 Ronald A DeVore and Robert C Sharpley. Besov spaces on domains in  $\mathbb{R}^d$ . *Transactions of the American Mathematical Society*, 335(2):843–864, 1993.
- 23 L. Devroye and L. Györfi. *Nonparametric Density Estimation: The  $L_1$  View*. John Wiley & Sons, 1985.
- 24 Luc Devroye and Gábor Lugosi. *Combinatorial methods in density estimation*. Springer Science & Business Media, 2012.
- 25 I. Diakonikolas, D. M. Kane, and A. Stewart. Efficient Robust Proper Learning of Log-concave Distributions. *CoRR*, abs/1606.03077, 2016.
- 26 I. Diakonikolas, D. M. Kane, and A. Stewart. Optimal learning via the Fourier transform for sums of independent integer random variables. In *COLT*, pages 831–849, 2016.
- 27 I. Diakonikolas, D. M. Kane, and A. Stewart. The Fourier transform of Poisson multinomial distributions and its algorithmic applications. In *STOC*, pages 1060–1073, 2016.
- 28 Ilias Diakonikolas, Daniel M Kane, and Jelani Nelson. Bounded independence fools degree-2 threshold functions. In *FOCS*, 2010.
- 29 Ilias Diakonikolas, Daniel M Kane, and Alistair Stewart. Learning multivariate log-concave distributions. In *Conference on Learning Theory (COLT)*, pages 711–727, 2016.

- 30 Lasse Holmström and Jussi Klemelä. Asymptotic bounds for the expected L1 error of a multivariate kernel density estimator. *Journal of multivariate analysis*, 42(2):245–266, 1992.
- 31 Il'dar Abdullovich Ibragimov. On the composition of unimodal distributions. *Theory of Probability & Its Applications*, 1(2):255–260, 1956.
- 32 S. Johnson. Saddle-point integration of C-infinity “bump” functions. arXiv preprint, 2015. [arXiv:1508.04376](https://arxiv.org/abs/1508.04376).
- 33 A. T. Kalai, A. Moitra, and G. Valiant. Efficiently learning mixtures of two Gaussians. In *STOC*, pages 553–562, 2010.
- 34 Daniel M Kane, Jelani Nelson, and David P Woodruff. On the exact space complexity of sketching and streaming small norms. In *SODA*, 2010.
- 35 A.K.H. Kim and R.J. Samworth. Global rates of convergence in log-concave density estimation. Available at arXiv, 2014. [arXiv:1404.2298](https://arxiv.org/abs/1404.2298).
- 36 Jussi Klemelä. *Smoothing of Multivariate Data: Density Estimation and Visualization*. Wiley Publishing, 2009.
- 37 László Lovász and Santosh Vempala. The geometry of logconcave functions and sampling algorithms. *Random Struct. Algorithms*, 30(3):307–358, 2007.
- 38 Elias Masry. Multivariate probability density estimation by wavelet methods: Strong consistency and rates for stationary time series. *Stochastic processes and their applications*, 67(2):177–193, 1997.
- 39 A. Moitra and G. Valiant. Settling the polynomial learnability of mixtures of Gaussians. In *FOCS*, pages 93–102, 2010.
- 40 A. Saumard and J.A. Wellner. Log-Concavity and Strong Log-Concavity: a review. Technical report, ArXiv, 23 April 2014. [arXiv:1404.5886](https://arxiv.org/abs/1404.5886).
- 41 Winthrop W Smith and Joanne M Smith. Handbook of real-time fast Fourier transforms. *IEEE, New York*, 1995.
- 42 Sergej Lvovich Sobolev. On a theorem of functional analysis. *Am. Math. Soc. Transl.*, 34:39–68, 1963.
- 43 Rebecca M Willett and Robert D Nowak. Multiscale Poisson intensity and density estimation. *IEEE Transactions on Information Theory*, 53(9):3171–3187, 2007.
- 44 Y. G. Yatracos. Rates of convergence of minimum distance estimators and Kolmogorov's entropy. *Annals of Statistics*, 13:768–774, 1985.

# From Local to Robust Testing via Agreement Testing

Irit Dinur<sup>1</sup>

Department of Mathematics and Computer Science, Weizmann Institute of Science,  
Rehovot, Israel  
irit.dinur@weizmann.ac.il

Prahladh Harsha<sup>2</sup>

Tata Institute of Fundamental Research, India  
prahladh@tifr.res.in

Tali Kaufman

Department of Computer Science, Bar-Ilan University, Ramat Gan, Israel  
kaufmant@mit.edu

Noga Ron-Zewi

Department of Computer Science, University of Haifa, Haifa, Israel  
noga@cs.haifa.ac.il

---

## Abstract

A local tester for an error-correcting code is a probabilistic procedure that queries a small subset of coordinates, accepts codewords with probability one, and rejects non-codewords with probability proportional to their distance from the code. The local tester is *robust* if for non-codewords it satisfies the stronger property that the average distance of local views from accepting views is proportional to the distance from the code. Robust testing is an important component in constructions of locally testable codes and probabilistically checkable proofs as it allows for composition of local tests.

In this work we show that for certain codes, any (natural) local tester can be converted to a robust tester with roughly the same number of queries. Our result holds for the class of *affine-invariant lifted codes* which is a broad class of codes that includes Reed-Muller codes, as well as recent constructions of high-rate locally testable codes (Guo, Kopparty, and Sudan, ITCS 2013). Instantiating this with known local testing results for lifted codes gives a more direct proof that improves some of the parameters of the main result of Guo, Haramaty, and Sudan (FOCS 2015), showing robustness of lifted codes.

To obtain the above transformation we relate the notions of local testing and robust testing to the notion of *agreement testing* that attempts to find out whether valid partial assignments can be stitched together to a global codeword. We first show that agreement testing implies robust testing, and then show that local testing implies agreement testing. Our proof is combinatorial, and is based on expansion / sampling properties of the collection of local views of local testers. Thus, it immediately applies to local testers of lifted codes that query random affine subspaces in  $\mathbb{F}_q^m$ , and moreover seems amenable to extension to other families of locally testable codes with expanding families of local views.

**2012 ACM Subject Classification** Theory of computation → Computational complexity and cryptography

---

<sup>1</sup> Supported by ERC-CoG grant number 772839.

<sup>2</sup> Research supported in part by the UGC-ISF grant and the Swarnajayanti Fellowship. Part of the work was done when the author was visiting the Weizmann Institute of Science.



**Keywords and phrases** Local testing, Robust testing, Agreement testing, Affine-invariant codes, Lifted codes

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2019.29

## 1 Introduction

Our main result shows a transformation from *local testing* to *robust testing* for the class of *affine-invariant lifted codes*. We start by describing the notions of local testing, robust testing, and lifted codes.

### 1.1 Local testing and robust testing

A code is a subset  $C \subseteq \Sigma^n$ . The elements of  $C$  are called *codewords*,  $\Sigma$  is the *alphabet* of the code, and  $n$  is the *block length*. The *rate* of the code is the ratio  $(\log_{|\Sigma|} |C|)/n$ . The code is *linear* if  $\Sigma = \mathbb{F}_q$  where  $\mathbb{F}_q$  is the finite field of  $q$  elements, and  $C$  is an  $\mathbb{F}_q$ -linear subspace of  $\mathbb{F}_q^n$ . It will be convenient to think of codewords in  $C$  as functions  $f : U \rightarrow \Sigma$  where  $U$  is a domain of size  $n$ . For a pair of functions  $f, g : U \rightarrow \Sigma$  we let  $\text{dist}(f, g)$  denote the fraction of inputs  $x \in U$  for which  $f(x) \neq g(x)$ . The *relative distance*  $\text{dist}(C)$  of the code is the minimum of  $\text{dist}(f, g)$  over all codewords  $f, g \in C$ . For a function  $f : U \rightarrow \Sigma$  we let  $\text{dist}(f, C)$  denote the minimum of  $\text{dist}(f, g)$  over all codewords  $g \in C$ .

A *local tester* for the code  $C$  is a probabilistic oracle algorithm that on oracle access to a function  $f : U \rightarrow \Sigma$  makes at most  $Q$  queries to  $f$ , and accepts  $f \in C$  with probability one, while rejecting  $f \notin C$  with probability at least  $\alpha \cdot \text{dist}(f, C)$ . We refer to  $Q$  as the *query complexity* of the tester, and to  $\alpha$  as the *soundness*. In this work we shall restrict our attention to local testers that pick a random subset  $K \subseteq U$  of cardinality  $Q$  according to some distribution, and accept if and only if  $f|_K \in C|_K$ .<sup>3</sup> The requirement then is that  $f|_K \in C|_K$  with probability one whenever  $f \in C$ , and

$$\Pr_K[f|_K \notin C|_K] \geq \alpha \cdot \text{dist}(f, C) \quad (1)$$

otherwise.

In this work we will be interested in the stronger notion of robustness. We say that a local tester as above is *robust* if for non-codewords the average distance of its local views from accepting views is proportional to the distance of the given function from the code. That is, as before we require that  $f|_K \in C|_K$  with probability one whenever  $f \in C$ , but instead of (1) we now require that

$$\mathbb{E}_K[\text{dist}(f|_K, C|_K)] \geq \alpha \cdot \text{dist}(f, C) \quad (2)$$

whenever  $f \notin C$ . Here we refer to  $\alpha$  as the *robustness* of the tester.

The notion of robustness was introduced by Ben-Sasson and Sudan [8] based on analogous notions for probabilistically checkable proofs [5, 15]. Robustness is a natural property of local testers that relates the global distance of a function from the code to its local distance from accepting views on local views. Moreover, robustness is also an important ingredient in constructions of locally testable codes and probabilistically checkable proofs as it allows for composition of local tests. Specifically, it follows by definition that if a code  $C$  is robustly

<sup>3</sup> Local testers may generally apply a more complex predicate on  $f|_K$ . However, natural local testers are typically of the restricted form we consider, and moreover it can be shown that a local tester for a linear code must be of this form [6].

testable with query complexity  $Q$  and soundness  $\alpha$ , and additionally each local restriction  $C|_K$  is locally testable with query complexity  $Q'$  and soundness  $\alpha'$ , then the code  $C$  is locally testable with query complexity  $Q'$  and soundness  $\alpha \cdot \alpha'$ . This property is useful when local restrictions can be tested efficiently which can happen if the code has many symmetries (as is the case with the class of lifted codes considered in this work) or can be achieved, in the case of probabilistically checkable proof, by attaching a short proof of proximity.

One can easily observe that (2) implies (1) since  $f|_K \notin C|_K$  whenever  $\text{dist}(f|_K, C|_K) > 0$ , so robustness is a stronger requirement than local testing. For the other direction, note that a local tester with soundness  $\alpha$  has robustness at least  $\alpha/Q$  since  $\text{dist}(f|_K, C|_K) \geq 1/Q$  whenever  $f|_K \notin C|_K$ . A natural question is whether this loss in robustness is necessary, and whether robustness is strictly stronger notion than local testing. In this work we shall show that this loss is unnecessary for the class of lifted codes, discussed below.

## 1.2 Lifted codes

Lifted codes are specified by a *base code*  $C \subseteq \{\mathbb{F}_q^\ell \rightarrow \mathbb{F}_q\}$  and a dimension  $m \geq \ell$ . We further assume that the base code  $C$  is linear and *affine-invariant*, that is, for any codeword  $f \in C$ , and for any affine transformation  $A : \mathbb{F}_q^\ell \rightarrow \mathbb{F}_q^\ell$  it holds that  $f \circ A \in C$ . Given these we define the *lifted code*  $C^{\ell \nearrow m}$  to be the code consisting of all functions  $f : \mathbb{F}_q^m \rightarrow \mathbb{F}_q$  that satisfy that  $f|_L \in C$  for any  $\ell$ -dimensional affine subspace  $L$ .

Lifted codes were first introduced by Ben-Sasson et al [7], and their local testability properties were further explored in subsequent work [20, 21, 19]. They are a natural generalization of the well-studied family of Reed-Muller codes, and moreover they also give rise to new families of locally testable codes that outperform Reed-Muller codes in certain range of parameters [20]. Specifically, lifted codes lead to one of the two known constructions (the other one being tensor codes [8, 9, 27, 24]) of high-rate locally testable codes (i.e., locally testable codes with rate approaching one and sublinear locality). Generally, lifted codes form a natural subclass of affine-invariant codes satisfying the “single-orbit characterization” property that is known to imply local testability, as well as local decodability [23].

There is a natural local test associated with lifted codes: on oracle access to a function  $f : \mathbb{F}_q^m \rightarrow \mathbb{F}_q$ , pick a uniform random  $\ell$ -dimensional affine subspace  $L$  and accept if and only if  $f|_L \in C$ . It follows immediately by definition that this test accepts any valid codeword  $f \in C^{\ell \nearrow m}$  with probability one, but more work is required to show that this test is sound. Specifically, since the test forms a single orbit characterization, it follows from [23] that it has soundness roughly  $q^{-2\ell}$ . The dependence of the soundness on the dimension  $\ell$  was later eliminated in [21] who showed soundness that is only a function of  $q$  (though an extremely quickly decaying one).

As for robustness, the above local testing results, together with the straightforward transformation from local testing to robust testing, immediately give robustness that is dependent on the dimension  $\ell$ . This was eliminated recently in [19] who showed robustness of the form  $\text{poly}(\delta)$  (about  $\delta^{74}$ , where  $\delta$  is the relative distance of the code) for the local test that queries subspaces of slightly larger dimension of  $2\ell$ . Interestingly, [19] did not rely on the aforementioned local testing results, but rather relied on viewing lifted codes as the intersection of “modified tensor codes”. They then proceeded by showing that these modified tensor codes are robustly testable (using the proof method of [27] showing robustness of tensor codes), and that this implies local testability of the lifted code (see Section 2.4 for more details about the proof method of [19]).

### 1.3 Our results

Our main result gives a transformation from local testing to robust testing, that does not suffer the factor of  $Q$  (the query complexity) loss in robustness, for the class of lifted codes. The transformation uses local testability in a “black-box” manner, and shows that if a code in this family is locally testable (using the natural subspace tester) then it is also robustly testable with roughly the same number of queries and robustness.

For  $k \geq \ell$ , let the  $k$ -dimensional (subspace) test denote the local tester that on oracle access to a function  $f : \mathbb{F}_q^m \rightarrow \mathbb{F}_q$  queries a uniform random  $k$ -dimensional affine subspace  $K$  and accepts if and only if  $f|_K \in C^{\ell \nearrow k}$ .

► **Theorem 1 (Main).** *Let  $C \subseteq \{\mathbb{F}_q^\ell \rightarrow \mathbb{F}_q\}$  be an affine-invariant linear code, and  $m \geq k \geq \ell$ . Suppose that  $C^{\ell \nearrow m}$  is locally testable using the  $k$ -dimensional test with query complexity  $q^k$  and soundness  $\alpha$ , and let  $\delta := \min_{k \leq r \leq m} \text{dist}(C^{\ell \nearrow r})$ . Then  $C^{\ell \nearrow m}$  is robustly testable using the  $(2k + \log_q(4/\delta))$ -dimensional test with query complexity  $O(q^{2k}/\delta)$  and robustness  $\Omega(\alpha \cdot \delta^3)$ .*

Note that if the relative distance  $\delta$  is constant, we only incur a constant multiplicative loss in robustness and testing dimension.

To apply the above theorem one can instantiate it with the local testing result of [23] that says that lifted codes are locally testable using the  $\ell$ -dimensional test with soundness  $\approx q^{-2\ell}$  (see Theorem 6 below). However, to obtain constant robustness we need that the soundness of the initial local tester would be constant (independent of  $q$  and  $\ell$ ), and for this we observe (in Proposition 19) that the soundness of [23] can be easily amplified to  $\Omega(1)$  at the cost of increasing the testing dimension to  $\approx 3\ell$ .<sup>4</sup> Using this observation we obtain the following.

► **Corollary 2.** *Let  $C \subseteq \{\mathbb{F}_q^\ell \rightarrow \mathbb{F}_q\}$  be an affine-invariant linear code of relative distance  $\delta$ , and  $m \geq \ell$ . Then  $C^{\ell \nearrow m}$  is robustly testable using the  $(6\ell + \log_q(128/\delta))$ -dimensional test with robustness  $\Omega(\delta^3)$ .*

Compared to the above corollary, [19] use lower dimension of  $2\ell$ , but also obtain lower soundness of  $\Omega(\delta^{74})$ .

As described next, our proof is combinatorial, relying mainly on expansion / sampling properties of the collection of local views. In particular, it uses very little about the algebraic structure of lifted codes or the base code. We thus hope that such techniques would prove useful in the future for showing robustness for other families of locally testable codes with similar expansion properties.

## 2 Proof overview

Our proof is based on a new connection between the notions of local testing, robust testing, and *agreement testing*. Specifically, we show that for the class of lifted codes agreement testing implies robust testing, and local testing implies agreement testing. The combination of these two implications gives our main Theorem 1. Next we elaborate on the notion of agreement testing, followed by an overview of each of the implications.

<sup>4</sup> Such an amplification with similar blow-up in query complexity can be easily obtained by repeating the test and accepting if and only if all invocations accept; we however need that the tester would be a subspace tester which can be obtained using sampling properties of affine subspaces.



## 2.1 Agreement testing

An *agreement test* attempts to find out whether partial assignments to local views can be stitched together to a single global codeword. Let  $C \subseteq \{U \rightarrow \Sigma\}$  be a code, and let  $\mathcal{S}$  be a collection of subsets of  $U$ . An *agreement tester* for  $C, \mathcal{S}$  is a probabilistic oracle algorithm that receives oracle access to a collection of partial assignments  $\{f_S : S \rightarrow \Sigma \mid S \in \mathcal{S}\}$  on sets of  $\mathcal{S}$ , where  $f_S \in C|_S$  for any  $S \in \mathcal{S}$ . The tester queries a few of the  $f_S$ 's, and is required to accept with probability one any collection  $(f_S)_S$  that is consistent with some global codeword  $g \in C$  (that is,  $g|_S = f_S$  for any  $S \in \mathcal{S}$ ), while rejecting any inconsistent collection  $(f_S)_S$  with probability proportional to the minimal fraction of  $f_S$ 's that must be changed in order to be consistent with some global codeword. In this work we focus on the two query agreement tester that picks a pair of sets  $S, S' \in \mathcal{S}$  according to some distribution and accepts if and only if  $f_S$  and  $f_{S'}$  agree on their intersection  $S \cap S'$ .

Agreement testing has first appeared in PCP constructions [3, 2] as so-called “low degree tests”, and is a key component in the proof of almost all PCP theorems. A prime example is the line vs. line low degree test [17, 26] in the proof of the PCP theorem. In the PCP construction, a function on a large vector space is replaced by an ensemble of (supposed) restrictions to all possible affine lines. These restrictions are supplied by a prover and are not a priori guaranteed to agree with any single global function. The “low degree test” is run by the verifier to check that restrictions on intersecting lines agree with each other, i.e. they give the same value to the point of intersection. The main point of the argument is to show that the passing of the test implies agreement with a single global function. In these early low degree tests (including the linearity testing work of [10]) an agreement test component can be discerned but quite implicitly. Indeed, it was only separated in the works [25, 4] that looked at the so-called list-decoding regime<sup>5</sup>, with the goal of proving a large gap for the PCP.

Goldreich and Safra [18] tried to separate the algebraic aspect of the low degree test from the combinatorial, and formulated a more general “consistency test” which is also referred to as an agreement test. They also proved a certain local to global result which was too weak to be useful for PCPs. In hindsight it is clear that since their family of subsets consisted of axis parallel lines, the expansion was not strong enough for a good agreement test. Only recently [13] the role of expansion underlying the family of subsets had begun to be uncovered.

Work on agreement testing then continued the combinatorial direction of [18] mainly in the list-decoding regime for direct product testing [15, 12, 22, 16, 14]. The techniques developed in this line of work turn out to be useful also for agreement testing in the unique-decoding regime (which is the more standard testing regime), and in particular for our work here.

## 2.2 Agreement testing implies robust testing

We begin with an overview of the simpler implication from agreement testing to robust testing. Suppose that we have a two query agreement tester for  $C, \mathcal{S}$  as described above. We would like to show that the local tester that queries a random  $S \in \mathcal{S}$  is robust. Let  $\mathcal{T}$  be the collection of subsets of  $U$  formed by pairwise intersections of sets in  $\mathcal{S}$ . The main properties we need out of  $\mathcal{S}, \mathcal{T}$  are sampling properties, specifically, that  $\mathcal{S}$  samples well the set of points  $U$ , and that for any  $S \in \mathcal{S}$  all sets in  $\mathcal{T}$  contained in  $S$  sample well the set of points in  $S$ . The main property we need out of the code is that its restrictions to sets in  $\mathcal{T}$

<sup>5</sup> In the list decoding regime one would like to reject a function that is  $(1 - \epsilon)$ -far from the code with very high-probability of  $1 - O(\epsilon)$ .

have distance. In the case of lifted codes these properties can be guaranteed by letting  $\mathcal{S}, \mathcal{T}$  be families of affine subspaces of fixed dimension.

To see that the proposed local tester is indeed robust, suppose that we have a function  $f : U \rightarrow \Sigma$  that is close to  $C|_S$  on a typical  $S$ , our goal is to show that  $f$  is close to a codeword  $g \in C$ . We first create an instance  $(f_S)_S$  for the agreement tester by letting  $f_S \in C|_S$  be the closest valid assignment to  $f|_S$ . Next observe that since  $f|_S$  is typically close to  $f_S$ , and by assumption that  $T$ 's sample well inside  $S$ 's, for a typical  $T$  and  $S, S'$  containing  $T$  it holds that  $f_S|_T \approx f|_T \approx f_{S'}|_T$ , and by distance property on  $\mathcal{T}$  this implies in turn that typically  $f_S|_T = f_{S'}|_T$ . Consequently, agreement testability implies the existence of a codeword  $g \in C$  that agrees with most  $f_S$ , and so  $g|_S = f_S \approx f|_S$  for most  $S$ . But since  $\mathcal{S}$  samples well inside  $U$  we conclude that  $f$  must be close to  $g$  as required.

### 2.3 Local testing implies agreement testing

We now turn to the local testing to agreement testing implication which is a bit more involved. Suppose that we have a local testing algorithm for  $C$  that queries a random set  $K \in \mathcal{K}$  and accepts if and only if  $f|_K \in C|_K$ . We would like to obtain an agreement tester for  $C$  with respect to some collection of subsets  $\mathcal{S}$ . As before, let  $\mathcal{T}$  be the collection of subsets of  $U$  formed by pairwise intersections of sets in  $\mathcal{S}$ . Once more the main properties we require out of  $\mathcal{S}, \mathcal{T}, \mathcal{K}$  are sampling properties. Specifically, we need that  $\mathcal{S}$  samples well inside  $U$ , and that for any  $T \in \mathcal{T}$  all sets in  $\mathcal{K}$  contained in  $T$  sample well inside  $T$ . We also require distance properties out of  $C$ , specifically that  $C$  has distance on  $U$  and on restrictions to sets in  $\mathcal{S}$  and  $\mathcal{T}$ . Once more, in the case of lifted codes these properties can be guaranteed by letting  $\mathcal{S}, \mathcal{T}, \mathcal{K}$  be families of affine subspaces of fixed dimension.

To show agreement testability, let  $(f_S)_S$  be a collection of valid assignments to sets in  $\mathcal{S}$  (so  $f_S \in C|_S$  for any  $S$ ), and suppose that  $f_S$  agrees with  $f_{S'}$  on  $S \cap S'$  for most pairs  $S, S'$ . Our goal will be to find a global codeword  $g \in C$  that agrees with most  $f_S$ . We find the function  $g$  in the following three stages.

#### Initial stage

In the first stage we define for any  $K \in \mathcal{K}$  a “most popular function”  $\text{Plur}_K$  by choosing the most common value among  $f_S|_K$  going over all  $S \in \mathcal{S}$  containing  $K$ . We then show, using the assumption that  $f_S$ 's typically agree on their intersections, that this most popular function is obtained with overwhelming probability for a typical  $K$ .

#### Local structure stage

In the second stage we define for each  $K \in \mathcal{K}$  a function  $g_K : U \rightarrow \Sigma$  by letting  $g_K(x)$  be the most common “vote” among all  $f_S$  that contain  $K$  and  $x$  and agree with  $\text{Plur}_K$  on  $K$  (this function is well defined because of the initial stage). We then show that for a typical  $K$ ,  $g_K$  is close to some function  $h_K \in C$ , and moreover  $h_K|_S = f_S$  for most  $S$  containing  $K$ .

To see why the above holds, first note that by assumptions that  $C$  has distance on  $T$ 's, and  $K$ 's sample well inside  $T$ 's, if a pair of  $f_S$ 's agree on  $K$  then they must typically also agree on their whole intersection. Therefore  $g_K(x)$  is also typically defined with overwhelming probability. Consequently, for a typical  $K$ , and most  $K'$ ,  $g_K|_{K'}$  agrees with some  $f_S$ . Recalling that  $f_S$ 's are valid assignments, local testability then implies the existence of  $h_K \in C$  that is close to  $g_K$ . The fact that  $h_K|_S = f_S$  for most  $S$  containing  $K$  follows by assumption that  $\mathcal{S}$  samples well  $U$ , and distance property on  $\mathcal{S}$ .

### Global structure stage

In the final stage we show that there exists  $\hat{K}$  such that  $h_{\hat{K}}$  agrees with  $f_S$  for most  $S$  (not necessarily containing  $\hat{K}$ ). We can then set our “global function”  $g$  to be equal to  $h_{\hat{K}}$ . To this end, we first observe that it suffices to show that most functions  $h_K$  are in fact identical. This now follows since for typical  $S \supseteq K \cup K'$  it holds that  $h_K|_S = f_S = h_{K'}|_S$ , and consequently since  $\mathcal{S}$  samples  $U$  it must typically hold that  $h_K = h_{K'}$ .

## 2.4 The proof method of Guo et al

The proof method of Guo et al [19] for showing robustness of lifted codes is very different from ours. In particular, it relies heavily on the algebraic structure of lifted codes. More specifically, the proof is based on viewing the lifted codes as the intersection of “modified tensor codes”. The *tensor product*  $C^{\otimes m}$  of a code  $C \subseteq \{\mathbb{F}_q \rightarrow \mathbb{F}_q\}$  can be thought of as the ‘axis-parallel lifting’ of  $C$ , that is, it is the code that consists of all functions  $f : \mathbb{F}_q^m \rightarrow \mathbb{F}_q$  whose restrictions to any axis-parallel line belong to  $C$ . The “modified tensor code” is a code of the form  $C_b^{\otimes m}$  where  $b$  is a direction in  $\mathbb{F}_q^m$ , and  $C_b^{\otimes m}$  consists of all functions  $f \in C^{\otimes m}$  whose restrictions to lines in direction  $b$  also belong to  $C$ .

The authors first use the proof method of [27], showing robust testing of tensor codes, to show that the modified tensor codes are also robustly testable. They then use the fact that the lifted code is the intersection of all codes of the form  $C_b^{\otimes m}$  for all directions  $b$  (this is true when the dimension of the base code for lifting is  $\ell = 1$ ; when  $\ell > 1$  the proof becomes more complicated) to deduce robust testability for the lifted code. However, since intersection of robustly testable codes is not necessarily robustly testable, a non-trivial work is required to show robust testability, which in particular exploits the degree structure of affine-invariant lifted codes.

The above program can be carried out only when the dimension  $m$  of the lifted code is a small constant multiple of  $\ell$ , and the authors use the “bootstrapping” technique [26, 2, 4, 1] to extend the result to work for arbitrary large  $m$ .

In contrast, we work directly with lifted codes of large dimension which allows us to exploit the sampling / expansion properties of large affine subspaces in  $\mathbb{F}_q^m$ . To the best of our knowledge, even for the special case of low-degree polynomials, this gives the first analysis of robustness that is not based on the two step approach of first analyzing the constant dimensional case and only then moving to the general dimensional case.

As opposed to [19] who reprove local testability on the way, our proof uses local testability in a black-box manner. Thus, it exhibits a separation between the algebraic properties that are used for showing local testability, and the combinatorial properties that are needed in order to turn local testability into robust testability.

### Paper organization

The rest of the paper is organized as follows. In Section 3 we set some notation, provide some definitions, and present the expansion properties of subspaces that we use. The transformation from agreement testing to robust testing is given in Section 4, while the transformation from local testing to agreement testing appears in Section 5. We wrap-up in Section 6 with the full transformation from local testing to robust testing that proves our main Theorem 1 and Corollary 2.

### 3 Preliminaries

For a prime power  $q$ , let  $\mathbb{F}_q$  denote the finite field of  $q$  elements. Let  $\{\mathbb{F}_q^m \rightarrow \mathbb{F}_q\}$  denote the set of functions mapping  $\mathbb{F}_q^m$  to  $\mathbb{F}_q$ . In what follows we focus on codes which are subsets of functions  $C \subseteq \{\mathbb{F}_q^m \rightarrow \mathbb{F}_q\}$ . For a pair of functions  $f, g : \mathbb{F}_q^m \rightarrow \mathbb{F}_q$  we use  $\text{dist}(f, g)$  to denote the fraction of inputs  $x \in \mathbb{F}_q^m$  for which  $f(x) \neq g(x)$ . The *relative distance*  $\text{dist}(C)$  of the code  $C$  is  $\min_{f \neq g \in C} \{\text{dist}(f, g)\}$ . For a function  $f : \mathbb{F}_q^m \rightarrow \mathbb{F}_q$  we use  $\text{dist}(f, C)$  to denote  $\min_{g \in C} \{\text{dist}(f, g)\}$ .

The code  $C$  is said to be *linear* if it is an  $\mathbb{F}_q$ -linear subspace, i.e., for every  $\alpha \in \mathbb{F}_q$  and  $f, g \in C$ , we have  $\alpha f + g \in C$ . A function  $A : \mathbb{F}_q^m \rightarrow \mathbb{F}_q^m$  is said to be an *affine transformation* if there exist a matrix  $M \in \mathbb{F}_q^{m \times m}$  and a vector  $b \in \mathbb{F}_q^m$  such that  $A(x) = Mx + b$ . The code  $C$  is said to be *affine-invariant* if for every affine transformation  $A$  and every  $f \in C$  we have  $f \circ A \in C$  (where  $(f \circ A)(x) = f(A(x))$ ).

#### 3.1 Lifted codes

A subset  $L \subseteq \mathbb{F}_q^m$  is said to be an  $\ell$ -dimensional affine subspace if there exist  $\alpha_0 \in \mathbb{F}_q^m$  and linearly independent  $\alpha_1, \dots, \alpha_\ell \in \mathbb{F}_q^m$  such that  $L = \{\alpha_0 + \sum_{i=1}^{\ell} \alpha_i x_i \mid x_1, \dots, x_\ell \in \mathbb{F}_q\}$ . We fix an arbitrary affine map  $\gamma_L : \mathbb{F}_q^\ell \rightarrow L$  (which we can view as a parameterization of  $L$ ). For a function  $f : \mathbb{F}_q^m \rightarrow \mathbb{F}_q$ , the restriction  $f|_L$  is viewed as a function in  $\{\mathbb{F}_q^\ell \rightarrow \mathbb{F}_q\}$  through  $f \circ \gamma_L : \mathbb{F}_q^\ell \rightarrow \mathbb{F}_q$ . In particular, when we ask if  $f|_L \stackrel{?}{\in} C \subseteq \{\mathbb{F}_q^\ell \rightarrow \mathbb{F}_q\}$  what we are really asking is whether  $f \circ \gamma_L \in C$ . Note that if  $C$  is affine-invariant, whether  $f|_L \in C$  does not depend on the choice of the parametrization  $\gamma_L$ .

► **Definition 3** (Lifted codes). Let  $C \subseteq \{\mathbb{F}_q^\ell \rightarrow \mathbb{F}_q\}$  be an affine-invariant linear code, and  $m \geq \ell$ . The  $m$ -dimensional lift  $C^{\ell \nearrow m}$  of  $C$  is given by

$$C^{\ell \nearrow m} := \{f : \mathbb{F}_q^m \rightarrow \mathbb{F}_q \mid f|_L \in C \text{ for every } \ell\text{-dimensional affine subspace } L \subseteq \mathbb{F}_q^m\}.$$

► **Proposition 4** (Distance of lifted codes, [20], Theorem 5.1, Part (2)). Let  $C \subseteq \{\mathbb{F}_q^\ell \rightarrow \mathbb{F}_q\}$  be an affine-invariant linear code, and  $m \geq \ell$ . Then  $\text{dist}(C^{\ell \nearrow m}) \geq \text{dist}(C) - q^{-\ell}$ .

#### 3.2 Local testing, robust testing, and agreement testing

We now formally define the notions of local testing, robust testing, and agreement testing, specialized to the class of lifted codes and subspace testers. In the case of local testing and robust testing this simply means that the tester samples a uniform random  $k$ -dimensional affine subspace and its accepting views are codewords in  $C^{\ell \nearrow k}$ .

► **Definition 5** (Local testing of lifted codes). Let  $C \subseteq \{\mathbb{F}_q^\ell \rightarrow \mathbb{F}_q\}$  be an affine-invariant linear code, and  $m \geq k \geq \ell$ . The  $m$ -dimensional lift  $C^{\ell \nearrow m}$  is  $(k, \alpha)$ -testable if for every  $f : \mathbb{F}_q^m \rightarrow \mathbb{F}_q$  it holds that

$$\Pr_K [f|_K \notin C^{\ell \nearrow k}] \geq \alpha \cdot \text{dist}(f, C^{\ell \nearrow m}),$$

where the probability is over a uniform random  $k$ -dimensional affine subspace  $K \subseteq \mathbb{F}_q^m$ .

► **Theorem 6** ([23], Theorem 2.9). Let  $C \subseteq \{\mathbb{F}_q^\ell \rightarrow \mathbb{F}_q\}$  be an affine-invariant linear code, and  $m \geq \ell$ . Then the  $\ell$ -dimensional test rejects a function  $f : \mathbb{F}_q^m \rightarrow \mathbb{F}_q$  with probability at least  $\frac{1}{2} \cdot \min \{q^{-2\ell}, \text{dist}(f, C^{\ell \nearrow m})\}$ . In particular,  $C^{\ell \nearrow m}$  is  $(\ell, \frac{q^{-2\ell}}{2})$ -testable.

► **Definition 7** (Robust testing of lifted codes). Let  $C \subseteq \{\mathbb{F}_q^\ell \rightarrow \mathbb{F}_q\}$  be an affine-invariant linear code, and  $m \geq k \geq \ell$ . The  $m$ -dimensional lift  $C^{\ell \nearrow m}$  is  $(k, \alpha)$ -robust if for every  $f : \mathbb{F}_q^m \rightarrow \mathbb{F}_q$  it holds that

$$\mathbb{E}_K [\text{dist}(f|_K, C^{\ell \nearrow k})] \geq \alpha \cdot \text{dist}(f, C^{\ell \nearrow m}),$$

where the expectation is over a uniform random  $k$ -dimensional affine subspace  $K \subseteq \mathbb{F}_q^m$ .

We note the following easy implications.

► **Proposition 8.** Let  $C \subseteq \{\mathbb{F}_q^\ell \rightarrow \mathbb{F}_q\}$  be an affine-invariant linear code, and  $m \geq k \geq \ell$ . Then the following hold.

1. If  $C^{\ell \nearrow m}$  is  $(k, \alpha)$ -robust then it is  $(k, \alpha)$ -testable.
2. If  $C^{\ell \nearrow m}$  is  $(k, \alpha)$ -testable then it is  $(k, \alpha \cdot q^{-k})$ -robust.
3. If  $C^{\ell \nearrow m}$  is  $(k, \alpha)$ -testable then it is  $(r, \alpha)$ -testable for any  $k \leq r \leq m$ .
4. If  $C^{\ell \nearrow m}$  is  $(k, \alpha)$ -robust then it is  $(r, \alpha)$ -robust for any  $k \leq r \leq m$ .

**Proof.** Part (1) follows since  $f|_K \notin C^{\ell \nearrow k}$  whenever  $\text{dist}(f|_K, C^{\ell \nearrow k}) > 0$ , while Part (2) follows since  $\text{dist}(f|_K, C^{\ell \nearrow k}) \geq q^{-k}$  whenever  $f|_K \notin C^{\ell \nearrow k}$ .

Part (3) follows by observing that for a uniform random  $r$ -dimensional affine subspace  $R$ ,

$$\begin{aligned} \Pr_R [f|_R \notin C^{\ell \nearrow r}] &= \mathbb{E}_R [\mathbb{1}_{f|_R \notin C^{\ell \nearrow r}}] \\ &\geq \mathbb{E}_R \left[ \Pr_{K \subseteq R} [f|_K \notin C^{\ell \nearrow k}] \right] \\ &= \Pr_K [f|_K \notin C^{\ell \nearrow k}], \end{aligned}$$

where the inequality follows since  $f|_R \in C^{\ell \nearrow r}$  implies that  $f|_K \in C^{\ell \nearrow k}$  for any  $K$ .

Finally, Part (4) follows by letting  $f_R$  be the codeword in  $C^{\ell \nearrow r}$  that is closest to  $f|_R$ , and noting that

$$\begin{aligned} \mathbb{E}_R [\text{dist}(f|_R, C^{\ell \nearrow r})] &= \mathbb{E}_R [\text{dist}(f|_R, f_R)] \\ &= \mathbb{E}_R [\mathbb{E}_{K \subseteq R} [\text{dist}(f|_K, f_R|_K)]] \\ &\geq \mathbb{E}_R [\mathbb{E}_{K \subseteq R} [\text{dist}(f|_K, C^{\ell \nearrow k})]] \\ &= \mathbb{E}_K [\text{dist}(f|_K, C^{\ell \nearrow k})], \end{aligned}$$

where the inequality follows since  $f_R|_K \in C^{\ell \nearrow k}$  for any  $K$ . ◀

We now turn to the definition of agreement testing. The agreement testers we consider are two query testers that for  $t < s$ , sample a uniform random  $t$ -dimensional affine subspace  $T$ , and a pair of uniform random  $s$ -dimensional affine subspaces  $S, S'$  containing  $T$ , and accept if and only if  $f_S, f_{S'}$  agree on  $T$ .

For a code  $C \subseteq \{\mathbb{F}_q^m \rightarrow \mathbb{F}_q\}$  we let  $C(s)$  be the code containing all collections  $(f_S)_S$  of partial assignments to  $s$ -dimensional affine subspaces that are consistent with some global codeword  $g \in C$ , formally,

$$C(s) := \{(f_S)_S \mid \exists g \in C \text{ such that } g|_S = f_S \text{ for any } s\text{-dimensional affine subspace } S\}.$$

For a pair of collections  $(f_S)_S, (g_S)_S$  of partial assignments to  $s$ -dimensional affine subspaces we denote by  $\text{dist}((f_S)_S, (g_S)_S)$  the fraction of  $s$ -dimensional affine subspaces  $S$  for which  $f_S \neq g_S$ , and we define  $\text{dist}((f_S)_S, C(s))$  accordingly.

► **Definition 9** (Agreement testing of lifted codes). Let  $C \subseteq \{\mathbb{F}_q^\ell \rightarrow \mathbb{F}_q\}$  be an affine-invariant linear code, and  $m \geq s > t \geq \ell$ . The  $m$ -dimensional lift  $C^{\ell \nearrow m}$  is  $(s, t, \alpha)$ -agreement testable if for every collection  $(f_S)_S$  where  $f_S \in C^{\ell \nearrow s}$  for every  $s$ -dimensional affine subspace  $S$  it holds that

$$\Pr_{T, S \supseteq T, S' \supseteq T} [f_S|_T \neq f_{S'}|_T] \geq \alpha \cdot \text{dist}((f_S)_S, C^{\ell \nearrow m}(s)),$$

where the probability is over a uniform random  $t$ -dimensional affine subspace  $T \subseteq \mathbb{F}_q^m$ , and uniform random  $s$ -dimensional affine subspaces  $S, S'$  containing  $T$ .

### 3.3 Subspace expansion

Let  $d_0, d_1, d_2 \in \{0, 1, \dots, m\}$  be integers, and let  $W \subseteq \mathbb{F}_q^m$  be a fixed affine subspace of dimension  $d_0$ . We denote by  $\mathcal{I}_{d_1, d_2}(d_0)$  the bipartite graph whose left side are all  $d_1$ -dimensional affine subspaces of  $\mathbb{F}_q^m$ , whose right side are all  $d_2$ -dimensional affine subspaces of  $\mathbb{F}_q^m$  containing  $W$ , and an edge  $(U, V)$  is present in the graph if and only if  $U \subseteq V$  (note that the structure of the graph is independent of the choice of  $W$ ). Our proof makes use of expansion properties of this graph.

► **Proposition 10.** *The second largest normalized singular value of the adjacency matrix of  $\mathcal{I}_{d_1, d_2}(d_0)$  is at most  $q^{-(d_2 - d_1 - d_0)/2}$ .*

**Proof.** Let  $\text{Gr}(m, d_1)$  be the Grassmann graph whose vertices are  $d_1$ -dimensional spaces and edges connect two  $d_1$ -spaces that intersect on an  $d_1 - 1$  space. We quote [11, Theorem 9.3.3] that gives the un-normalized eigenvalues

$$\theta_j = q^{j+1} \begin{bmatrix} d_1 - j \\ 1 \end{bmatrix} \begin{bmatrix} n - d_1 - j \\ 1 \end{bmatrix} - \begin{bmatrix} j \\ 1 \end{bmatrix}$$

and the degree is

$$k = q \begin{bmatrix} d_1 \\ 1 \end{bmatrix} \begin{bmatrix} n - d_1 \\ 1 \end{bmatrix}$$

Plugging in  $j = 1$  one gets the second largest eigenvalue in absolute value is approximately

$$\lambda(\text{Gr}(m, d_1)) \approx \frac{1}{\sqrt{q}}. \quad (3)$$

It can be shown that  $\lambda(\mathcal{I}_{d_1, d_2}(0)) \approx (\lambda(\text{Gr}(m, d_1)))^{d_2 - d_1}$ . When adding  $W$  we are essentially moving to the graph  $\mathcal{I}_{d_1, d_2 - d_0}(0)$ , i.e.  $\lambda(\mathcal{I}_{d_1, d_2}(d_0)) \approx \lambda(\mathcal{I}_{d_1, d_2 - d_0}(0))$ . ◀

We shall use the following sampling property of  $\mathcal{I}_{d_1, d_2}(d_0)$ .

► **Proposition 11.** *Let  $G = (L \cup R, E)$  be a bipartite graph with second largest normalized singular value  $\lambda$ . Then for any subset  $A \subseteq L$  of density  $\alpha$  it holds that  $|N(A)| \geq (1 - \lambda^2/\alpha) \cdot |R|$  where  $N(A)$  denotes the set of neighbors of  $A$  in  $R$ .*

**Proof.** Let  $B := R \setminus N(A)$  and  $\beta := |B|/|R|$ . Noting that  $\Pr_{(u,v) \in E} [u \in A \wedge v \in B] = 0$ , by expander mixing lemma (see e.g., [13, Lemma 2.8.]) we have that

$$\alpha\beta = \left| \Pr_{(u,v) \in E} [u \in A \wedge v \in B] - \alpha\beta \right| \leq \lambda\sqrt{\alpha\beta},$$

and so  $\beta \leq \lambda^2/\alpha$ . It follows that  $|N(A)| = (1 - \beta)|R| \geq (1 - \lambda^2/\alpha)|R|$ . ◀

## 4 From agreement testing to robust testing

In this section we prove the following lemma showing the agreement testing to robust testing implication.

► **Lemma 12** (Agreement testing implies robust testing). *Let  $C \subseteq \{\mathbb{F}_q^\ell \rightarrow \mathbb{F}_q\}$  be an affine-invariant linear code, and  $m \geq s > t \geq \ell$ . Suppose that  $C^{\ell \nearrow m}$  is  $(s, t, \alpha)$ -agreement testable, and let  $\delta := \text{dist}(C^{\ell \nearrow t})$ . Then  $C^{\ell \nearrow m}$  is  $(s, \Omega(\alpha\delta))$ -robust.*

**Proof.** For simplicity of notation, in what follows we let  $T, S$  denote the random variables obtained by sampling a uniform random affine subspace of dimension  $t, s$  respectively. Suppose that  $f : \mathbb{F}_q^m \rightarrow \mathbb{F}_q$  has  $\mathbb{E}_S[\text{dist}(f|_S, C^{\ell \nearrow s})] \leq \epsilon$ , our goal is to find a codeword  $g \in C^{\ell \nearrow m}$  such that  $\text{dist}(f, g) \leq O(\epsilon/(\alpha\delta))$ .

The proof proceeds as follows. We would like to apply our assumption on agreement testability, and towards this, we create an instance  $(f_S)_S$  for the agreement tester by letting  $f_S$  be the codeword in  $C^{\ell \nearrow s}$  that is closest to  $f|_S$ . We then use the fact that  $f_S$  is typically close to  $f|_S$ , together with the fact that  $t$ -dimensional affine subspaces sample well inside  $s$ -dimensional affine subspaces, and the assumption that  $C$  has distance on  $t$ -dimensional affine subspaces, to show that  $\Pr_{T, S \supseteq T, S' \supseteq T}[f_S|_T \neq f_{S'}|_T]$  is small. Agreement testability then gives a codeword  $g \in C^{\ell \nearrow m}$  that is consistent with most  $f_S$ , and using the fact that  $s$ -dimensional affine subspaces sample well inside  $\mathbb{F}_q^m$  this implies in turn that  $\text{dist}(f, g)$  is small. Details follow.

We begin by showing that  $\Pr_{T, S \supseteq T, S' \supseteq T}[f_S|_T \neq f_{S'}|_T]$  is small. Recall first that  $\mathbb{E}_S[\text{dist}(f|_S, f_S)] = \mathbb{E}_S[\text{dist}(f|_S, C^{\ell \nearrow s})] \leq \epsilon$ . Next observe that for a fixed  $s$ -dimensional affine subspace  $S$ , any point in a uniform random  $t$ -dimensional affine subspace contained in  $S$  is uniform in  $S$ . Thus we also have that  $\mathbb{E}_{S, T \subseteq S}[\text{dist}(f|_T, f_S|_T)] \leq \epsilon$ , and consequently

$$\begin{aligned} \Pr_{T, S \supseteq T, S' \supseteq T}[\text{dist}(f_S|_T, f_{S'}|_T) \geq \delta] &\leq 2 \cdot \Pr_{T, S \supseteq T}[\text{dist}(f|_T, f_S|_T) \geq \delta/2] \\ &= 2 \cdot \Pr_{S, T \subseteq S}[\text{dist}(f|_T, f_S|_T) \geq \delta/2] \\ &\leq \frac{4\epsilon}{\delta}. \end{aligned}$$

But since  $f_S|_T, f_{S'}|_T$  are both codewords of  $C^{\ell \nearrow t}$ , a code of relative distance  $\delta$ , the above implies in turn that  $\Pr_{T, S \supseteq T, S' \supseteq T}[f_S|_T \neq f_{S'}|_T] \leq \frac{4\epsilon}{\delta}$ .

Our assumption on agreement testability now gives a codeword  $g \in C^{\ell \nearrow m}$  that has  $\Pr_S[g|_S \neq f_S] \leq 4\epsilon/(\alpha\delta)$ . But since any point in a uniform random  $s$ -dimensional affine subspace is uniform in  $\mathbb{F}_q^m$  this gives in turn that

$$\text{dist}(f, g) = \mathbb{E}_S[\text{dist}(f|_S, g|_S)] \leq \mathbb{E}_S[\text{dist}(f|_S, f_S)] + \mathbb{E}_S[\text{dist}(f_S, g|_S)] \leq \epsilon + \frac{4\epsilon}{\alpha\delta} \leq \frac{5\epsilon}{\alpha\delta}. \quad \blacktriangleleft$$

## 5 From local testing to agreement testing

In this section we prove the following lemma that gives the local testing to agreement testing implication.

► **Lemma 13** (Local testing implies agreement testing). *Let  $C \subseteq \{\mathbb{F}_q^\ell \rightarrow \mathbb{F}_q\}$  be an affine-invariant linear code, and  $m \geq k \geq \ell$ . Suppose that  $C^{\ell \nearrow m}$  is  $(k, \alpha)$ -testable, and let  $\delta := \min_{k \leq r \leq m} \text{dist}(C^{\ell \nearrow r})$ . Then  $C^{\ell \nearrow m}$  is  $(2k + \log_q(4/\delta), k + 1, \Omega(\alpha \cdot \delta^2))$ -agreement testable.*



**Proof outline**

For simplicity of notation, in what follows let  $s := 2k + \log_q(4/\delta)$  and  $t := k + 1$ . We let both  $S, S'$  ( $T, T'$  and  $K, K'$ , resp.) denote random variables obtained by sampling a uniform random affine subspace of dimension  $s$  ( $t, k$ , resp.).

Let  $(f_S)_S$  be a collection of partial assignments such that  $f_S \in C^{\ell, \gamma^s}$  for every  $S$ , and

$$\Pr_{T, S \supseteq T, S' \supseteq T} [f_S|_T \neq f_{S'}|_T] \leq \epsilon. \quad (4)$$

Our goal is to find a global codeword  $g \in C^{\ell, \gamma^m}$  that has

$$\Pr_S [g|_S \neq f_S] \leq O\left(\frac{\epsilon}{\alpha \cdot \delta^2}\right). \quad (5)$$

We find the codeword  $g$  in three stages.

1. In the initial stage (Section 5.1) we define for any  $k$ -dimensional affine subspace  $K$  a “most popular function”  $\text{Plur}_K : \mathbb{F}_q^k \rightarrow \mathbb{F}_q$  by choosing the most common value among  $f_S|_K$  going over all  $S \supseteq K$ . We show that for a typical  $K$ , this function is obtained with an overwhelming plurality of  $1 - O(\epsilon)$ .
2. In the “local structure” stage (Section 5.2) we define for any  $k$ -dimensional affine subspace  $K$  a function  $g_K : \mathbb{F}_q^m \rightarrow \mathbb{F}_q$  by letting  $g_K(x)$  be the most common “vote” among all  $f_S$  that contain both  $K$  and  $x$  and agree with  $\text{Plur}_K$  on  $K$ . We then show that for a typical  $K$ ,  $g_K$  is close to some codeword  $h_K \in C^{\ell, \gamma^m}$ , and moreover  $h_K|_S = f_S$  for most  $S$  containing  $K$ .
3. In the “global structure” stage (Section 5.3) we show that there exists  $\hat{K}$  for which  $h_{\hat{K}}|_S = f_S$  for most  $S$  (not necessarily containing  $\hat{K}$ ). We can then set our “global function”  $g$  to be equal to  $h_{\hat{K}}$ .

## 5.1 Initial stage

For any  $k$ -dimensional affine subspace  $K$  we let  $\text{Plur}_K : \mathbb{F}_q^k \rightarrow \mathbb{F}_q$  denote the most common value among  $f_S|_K$  for  $S$  containing  $K$ , that is,

$$\text{Plur}_K := \text{plurality}_{S \supseteq K} \{f_S|_K\}.$$

Next we use our assumption (4) to show that for a typical  $K$ , the function  $\text{Plur}_K$  is obtained with overwhelming plurality.

► **Lemma 14.**

$$\mathbb{E}_K \left[ \Pr_{S \supseteq K} [f_S|_K \neq \text{Plur}_K] \right] \leq 2\epsilon.$$

**Proof.** Since the collision probability lower bounds the probability of hitting the most common value, it suffices to show that

$$\Pr_{K, S \supseteq K, S' \supseteq K} [f_S|_K \neq f_{S'}|_K] \leq 2\epsilon. \quad (6)$$

Clearly if  $t = k$  we would be done by (4), so the whole point is to show the same for  $t > k$ . We describe a distribution on triples  $(S_1, S', S_2)$  such that  $(S_1, S_2)$  are distributed as in (6) but the pairs  $(S_1, S')$  and  $(S', S_2)$  are distributed as in (4):

1. Choose a uniform random  $k$ -dimensional affine subspace  $K$ .
2. Choose a pair of uniform random  $t$ -dimensional affine subspaces  $T_1, T_2$  containing  $K$ .

3. For  $i = 1, 2$ , choose a uniform random  $s$ -dimensional affine subspace  $S_i$  containing  $T_i$ .
4. Choose a uniform random  $s$ -dimensional affine subspace  $S'$  containing  $T_1 \cup T_2$  (this can be done since  $t = k + 1$  and  $s \geq k + 2$ ).

One can check that indeed  $K, S_1, S_2$  are distributed as in (6) while  $T_i, S_i, S'$  are distributed as in (4). Thus by our assumption (4),

$$\begin{aligned} & \Pr_{K, S_1 \supseteq K, S_2 \supseteq K} [f_{S_1}|_K \neq f_{S_2}|_K] \\ & \leq \Pr_{T_1, S_1 \supseteq T_1, S' \supseteq T_1} [f_{S_1}|_{T_1} \neq f_{S'}|_{T_1}] + \Pr_{T_2, S' \supseteq T_2, S_2 \supseteq T_2} [f_{S'}|_{T_2} \neq f_{S_2}|_{T_2}] \leq 2\epsilon. \quad \blacktriangleleft \end{aligned}$$

## 5.2 Local structure

Next we define for every  $k$ -dimensional affine subspace  $K$  the function  $g_K : \mathbb{F}_q^m \rightarrow \mathbb{F}_q$ . As described above, for every  $x \in \mathbb{F}_q^m$ , we let  $g_K(x)$  be the most common value among  $f_S(x)$  for  $S$  that contain both  $K$  and  $x$  and agree with  $\text{Plur}_K$  on  $K$ , that is,

$$g_K(x) := \text{plurality}_{S \supseteq K \cup \{x\}, f_S|_K = \text{Plur}_K} \{f_S(x)\}.$$

Next we would like to show that for a typical  $K$ ,  $g_K$  is close to some codeword  $h_K \in C^{\ell \nearrow m}$ , and additionally  $h_K|_S = f_S$  for most  $S$  containing  $K$ . We show these in three steps:

1. Boosting step (Lemma 15): In this step we show that for typical  $K, x$ , the plurality in the definition of  $g_K(x)$  is obtained with overwhelming probability.
2. LTC step (Lemma 16): In this step we use the previous step to show that for a typical  $g_K$ , for most  $K'$ ,  $g_K|_{K'}$  agrees with some  $f_S$  on  $K'$ , and therefore is a codeword of  $C^{\ell \nearrow k}$ . By local testability assumption this implies in turn that such  $g_K$  is close to being in the code  $C^{\ell \nearrow m}$ , and we denote by  $h_K \in C^{\ell \nearrow m}$  the closest codeword to  $g_K$ .
3. Agreement step (Lemma 17): In this step we show that a typical  $h_K$  agrees with most  $f_S$  for  $S \supseteq K$ .

We start with the boosting step, showing that for typical  $K, x$ , the plurality in the definition of  $g_K(x)$  is obtained with overwhelming probability. Intuitively, this follows by assumption that the code has distance on  $t$ -dimensional affine subspaces, together with the fact that  $k$ -dimensional affine subspaces sample well inside  $t$ -dimensional affine subspaces, which imply that if a pair of  $f_S$  agree on  $K$  then they must typically also agree on their whole intersection.

► **Lemma 15** (Boosting step).

$$\mathbb{E}_{K, x \notin K} \left[ \Pr_{S \supseteq K \cup \{x\}} [f_S(x) \neq g_K(x) \mid f_S|_K = \text{Plur}_K] \right] \leq O \left( q^{-k} \cdot \frac{\epsilon}{\delta \cdot (1 - 4\epsilon)} \right).$$

**Proof.** Since the collision probability lower bounds the probability of hitting the most common value, it suffices to show that

$$\Pr_{K, x \notin K, S \supseteq K \cup \{x\}, S' \supseteq K \cup \{x\}} [f_S(x) \neq f_{S'}(x) \mid f_S|_K = f_{S'}|_K = \text{Plur}_K] \leq O \left( q^{-k} \cdot \frac{\epsilon}{\delta \cdot (1 - 4\epsilon)} \right).$$

Now we have that

$$\begin{aligned} & \Pr_{K, x \notin K, S \supseteq K \cup \{x\}, S' \supseteq K \cup \{x\}} [f_S(x) \neq f_{S'}(x) \mid f_S|_K = f_{S'}|_K = \text{Plur}_K] \\ & \leq \Pr_{K, T \supseteq K, S \supseteq T, S' \supseteq T} [f_S|_T \neq f_{S'}|_T \mid f_S|_K = f_{S'}|_K = \text{Plur}_K] \\ & = \frac{\Pr_{T, S \supseteq T, S' \supseteq T, K \subseteq T} [f_S|_K = f_{S'}|_K = \text{Plur}_K \mid f_S|_T \neq f_{S'}|_T] \cdot \Pr_{T, S \supseteq T, S' \supseteq T} [f_S|_T \neq f_{S'}|_T]}{\Pr_{K, T \supseteq K, S \supseteq T, S' \supseteq T} [f_S|_K = f_{S'}|_K = \text{Plur}_K]}. \end{aligned}$$

## 29:14 From Local to Robust Testing via Agreement Testing

Next we bound each of the terms above.

By our assumption (4), the right hand term in the numerator is upper bounded by  $\epsilon$ . To bound the denominator note that by Lemma 14,

$$\Pr_{K, T \supseteq K, S \supseteq T, S' \supseteq T} [f_S|_K = f_{S'}|_K = \text{Plur}_K] \geq 1 - 2 \cdot \Pr_{K, S \supseteq K} [f_S|_K \neq \text{Plur}_K] \geq 1 - 4\epsilon. \quad (7)$$

To bound the left hand term in the numerator note first that since  $f_S, f_{S'}$  are both codewords of  $C^{\ell, \lambda^s}$  then  $f_S|_T, f_{S'}|_T$  are distinct codewords in  $C^{\ell, \lambda^t}$ , and so  $\text{dist}(f_S|_T, f_{S'}|_T) \geq \delta$ . We now apply Propositions 10 and 11 on the graph  $\mathcal{I}_{0,k}(0)$ : The ambient space is  $T$ , and the graph connects the points of  $T$  (which are the 0-dimensional affine subspaces contained in  $T$ ) on the left to the  $k$ -dimensional affine subspaces contained in  $T$  on the right. By Proposition 10 the graph  $\mathcal{I}_{0,k}(0)$  has second largest normalized singular value at most  $q^{-k/2}$ , and so taking  $A = \{x \in T \mid f_S(x) \neq f_{S'}(x)\}$  in Proposition 11 we deduce that at most  $q^{-k}/\delta$  fraction of  $K$  can miss  $A$  altogether. Thus,

$$\Pr_{T, S \supseteq T, S' \supseteq T, K \subseteq T} [f_S|_K = f_{S'}|_K \mid f_S|_T \neq f_{S'}|_T] \leq \frac{q^{-k}}{\delta}. \quad (8)$$

The final bound is obtained by combining the bounds in (4), (7), and (8).  $\blacktriangleleft$

Next we use the assumption on local testability to show that for a typical  $K$ ,  $g_K$  is close to being a codeword of  $C^{\ell, \lambda^m}$ .

► **Lemma 16** (LTC step).

$$\mathbb{E}_K [\text{dist}(g_K, C^{\ell, \lambda^m})] \leq O\left(\frac{\epsilon}{\alpha \cdot \delta}\right).$$

**Proof.** To apply our assumption on local testability we first show that  $g_K|_{K'}$  is typically a codeword of  $C^{\ell, \lambda^k}$ . For this, first observe that if  $g_K|_{K'}$  is not a codeword of  $C^{\ell, \lambda^k}$  then  $g_K|_{K'} \neq f_S|_{K'}$  for all  $S$  (since  $f_S \in C^{\ell, \lambda^s}$  and so  $f_S|_{K'} \in C^{\ell, \lambda^k}$ ). Thus we have

$$\begin{aligned} \mathbb{E}_{K, K'} [\mathbb{1}_{g_K|_{K'} \notin C^{\ell, \lambda^k}}] &\leq \mathbb{E}_{K, K'} \left[ \Pr_{S \supseteq K \cup K'} [g_K|_{K'} \neq f_S|_{K'}] \right] \\ &\leq \mathbb{E}_{K, K'} \left[ \Pr_{S \supseteq K \cup K'} [g_K|_{K'} \neq f_S|_{K'} \mid f_S|_K = \text{Plur}_K] \right] + \Pr_{K, S \supseteq K} [f_S|_K \neq \text{Plur}_K] \end{aligned}$$

We claim that the above expression is at most  $O(\epsilon/\delta)$ . To see this note first that by Lemma 14 the right hand term is at most  $2\epsilon$ . To bound the left hand term, note that since each individual point in  $K'$  is uniformly distributed in  $\mathbb{F}_q^m$  this term is upper bounded by

$$q^k \cdot \mathbb{E}_{K, x} \left[ \Pr_{S \supseteq K \cup \{x\}} [g_K(x) \neq f_S(x) \mid f_S|_K = \text{Plur}_K] \right],$$

which is in turn at most  $O(\epsilon/\delta)$  by Lemma 15 (noting that the probability in the above expression is zero whenever  $x \in K$ ).

Finally, for any  $k$ -dimensional affine subspace  $K$  let  $\epsilon_K := \Pr_{K'} [g_K|_{K'} \notin C^{\ell, \lambda^k}]$ . Then on the one hand  $\mathbb{E}_K[\epsilon_K] = \mathbb{E}_{K, K'} [\mathbb{1}_{g_K|_{K'} \notin C^{\ell, \lambda^k}}] \leq O(\epsilon/\delta)$ , and on the other hand  $\text{dist}(g_K, C^{\ell, \lambda^m}) \leq \epsilon_K/\alpha$  for any  $K$  by assumption that  $C^{\ell, \lambda^m}$  is  $(k, \alpha)$ -testable. We conclude that

$$\mathbb{E}_K [\text{dist}(g_K, C^{\ell, \lambda^m})] \leq \mathbb{E}_K \left[ \frac{\epsilon_K}{\alpha} \right] \leq O\left(\frac{\epsilon}{\alpha \cdot \delta}\right). \quad \blacktriangleleft$$

For any  $k$ -dimensional affine subspace  $K$  let  $h_K \in C^{\ell, \lambda^m}$  be the codeword that is closest to  $g_K$ . Then by the above lemma,

$$\mathbb{E}_K [\text{dist}(g_K, h_K)] \leq O\left(\frac{\epsilon}{\alpha \cdot \delta}\right).$$

The following lemma says that for a typical  $K$  we have that  $h_K|_S = f_S$  for most  $S$  containing  $K$ , which follows by the fact that  $s$ -dimensional affine subspaces sample well inside  $\mathbb{F}_q^m$  and by assumption that the code has distance on  $s$ -dimensional affine subspaces.

► **Lemma 17** (Agreement step).

$$\mathbb{E}_K \left[ \Pr_{S \supseteq K} [h_K|_S \neq f_S] \right] \leq O\left(\frac{\epsilon}{\alpha \cdot \delta^2}\right).$$

**Proof.** We show that for typical  $S \supseteq K$ , on the one hand, by Lemma 16 and the fact that  $s$ -dimensional affine subspaces sample well inside  $\mathbb{F}_q^m$ ,  $\text{dist}(h_K|_S, g_K|_S)$  is small, and on the other hand, by Lemma 15,  $\text{dist}(g_K|_S, f_S)$  is small. We then conclude by triangle inequality that  $\text{dist}(h_K|_S, f_S)$  is small, which implies in turn that  $h_K|_S = f_S$  by assumption that the code has distance on  $s$ -dimensional subspaces.

We start by showing that  $\text{dist}(h_K|_S, g_K|_S)$  is typically small. For this note that for a fixed  $k$ -dimensional affine subspace  $K$  and uniform random  $S$  containing  $K$ , each individual point in  $S \setminus K$  is uniformly distributed in  $\mathbb{F}_q^m \setminus K$ . Thus we have

$$\mathbb{E}_{K, S \supseteq K} [\text{dist}(h_K|_S, g_K|_S)] \leq q^{-(s-k)} + \mathbb{E}_K [\text{dist}(h_K, g_K)] \leq \frac{\delta}{4} + O\left(\frac{\epsilon}{\alpha \cdot \delta}\right), \quad (9)$$

where the last inequality follows by choice of  $s \geq k + \log_q(4/\delta)$  and Lemma 16.

Next we show that  $\text{dist}(h_K|_S, f_S)$  is typically small. For this note that

$$\begin{aligned} & \mathbb{E}_{K, S \supseteq K} [\text{dist}(g_K|_S, f_S)] \\ & \leq q^{-(s-k)} + \mathbb{E}_{K, x \notin K} \left[ \Pr_{S \supseteq K \cup \{x\}} [g_K(x) \neq f_S(x)] \right] \\ & \leq q^{-(s-k)} + \mathbb{E}_{K, x \notin K} \left[ \Pr_{S \supseteq K \cup \{x\}} [g_K(x) \neq f_S(x) \mid f_S|_K = \text{Plur}_K] \right] + \Pr_{K, S \supseteq K} [f_S|_K \neq \text{Plur}_K] \\ & \leq \frac{\delta}{4} + O\left(\frac{\epsilon}{\delta}\right), \end{aligned} \quad (10)$$

where the last inequality follows by choice of  $s \geq k + \log_q(4/\delta)$  and Lemmas 15 and 14.

Combining (9) and (10), by triangle inequality we have that

$$\mathbb{E}_{K, S \supseteq K} [\text{dist}(h_K|_S, f_S)] \leq \frac{\delta}{2} + O\left(\frac{\epsilon}{\alpha \cdot \delta}\right),$$

and by Markov's inequality,

$$\Pr_{K, S \supseteq K} [\text{dist}(h_K|_S, f_S) \geq \delta] \leq O\left(\frac{\epsilon}{\alpha \cdot \delta^2}\right).$$

Finally, since both  $h_K|_S$  and  $f_S$  are codewords of  $C^{\ell, \lambda^s}$  and  $\text{dist}(C^{\ell, \lambda^s}) \geq \delta$  we conclude that  $h_K|_S \neq f_S$  with probability at most  $O\left(\frac{\epsilon}{\alpha \cdot \delta^2}\right)$  over the choice of  $K$  and  $S \supseteq K$ . ◀

### 5.3 Global structure

We now complete the proof of Lemma 13 by showing that there exists a codeword  $g \in C^{\ell, \gamma^m}$  that agrees with most  $f_S$ . We start by showing that most functions  $h_K$  are in fact identical, which follows by Lemma 17 and the fact that  $s$ -dimensional affine subspaces sample well inside  $\mathbb{F}_q^m$ .

► **Lemma 18.** *There exists a  $k$ -dimensional affine subspace  $\hat{K}$  such that*

$$\Pr_K [h_K \neq h_{\hat{K}}] \leq O\left(\frac{\epsilon}{\alpha \cdot \delta^2}\right).$$

**Proof.** By Lemma 17,

$$\Pr_{K, K', S \supseteq K \cup K'} [h_K|_S \neq h_{K'}|_S] \leq 2 \cdot \Pr_{K, S \supseteq K} [h_K|_S \neq f_S] \leq O\left(\frac{\epsilon}{\alpha \cdot \delta^2}\right),$$

and so by averaging there exists  $\hat{K}$  such that

$$\Pr_{K, S \supseteq K \cup \hat{K}} [h_K|_S \neq h_{\hat{K}}|_S] \leq O\left(\frac{\epsilon}{\alpha \cdot \delta^2}\right).$$

Markov's inequality then implies that

$$\Pr_{S \supseteq K \cup \hat{K}} [h_K|_S \neq h_{\hat{K}}|_S] \geq \frac{1}{2}$$

with probability at most  $O\left(\frac{\epsilon}{\alpha \cdot \delta^2}\right)$  over the choice of  $K$ .

Next observe that if  $h_K \neq h_{\hat{K}}$  then since  $h_K, h_{\hat{K}}$  are both codewords of  $C^{\ell, \gamma^m}$  and  $\text{dist}(C^{\ell, \gamma^m}) \geq \delta$ , it must hold that  $\text{dist}(h_K, h_{\hat{K}}) \geq \delta$ . We now apply Propositions 10 and 11 on the graph  $\mathcal{I}_{0,s}(2k)$  that connects the points of  $\mathbb{F}_q^m$  on the left to the  $s$ -dimensional affine subspaces containing  $K \cup \hat{K}$  on the right. By Proposition 10 the graph  $\mathcal{I}_{0,s}(2k)$  has second largest normalized singular value at most  $q^{-(s-2k)/2}$ , and so taking  $A = \{x \in \mathbb{F}_q^m \mid h_K(x) \neq h_{\hat{K}}(x)\}$  in Proposition 11 we deduce that

$$\Pr_{S \supseteq K \cup \hat{K}} [h_{\hat{K}}|_S \neq h_K|_S] \geq 1 - \frac{q^{-(s-2k)}}{\delta} \geq 1/2,$$

where the last inequality follows by assumption that  $s \geq 2k + \log_q(2/\delta)$ .

It now follows that  $h_K \neq h_{\hat{K}}$  with probability at most  $O\left(\frac{\epsilon}{\alpha \cdot \delta^2}\right)$  over the choice of  $K$ . ◀

We can now complete the proof of Lemma 13.

**Proof of Lemma 13.** Set  $g \in C^{\ell, \gamma^m}$  to be the function  $h_{\hat{K}}$  guaranteed by Lemma 18. By Lemmas 17 and 18,

$$\Pr_S [g|_S \neq f_S] = \Pr_{K, S \supseteq K} [g|_S \neq f_S] \leq \Pr_K [g \neq h_K] + \Pr_{K, S \supseteq K} [h_K|_S \neq f_S] \leq O\left(\frac{\epsilon}{\alpha \cdot \delta^2}\right).$$

So  $g$  satisfies (5) as required. ◀

## 6 From local testing to robust testing

### 6.1 Proof of Main Theorem 1

We can now combine Lemmas 12 and 13 to prove our main Theorem 1, showing a transformation from local testing to robust testing.

**Proof of Theorem 1.** By Lemma 13 we have that  $C^{\ell, \gamma^m}$  is  $(2k + \log_q(4/\delta), k + 1, \Omega(\alpha \cdot \delta^2))$ -agreement testable, and by Lemma 12 this implies in turn that  $C^{\ell, \gamma^m}$  is  $(2k + \log_q(4/\delta), \Omega(\alpha \cdot \delta^3))$ -robust. ◀

## 6.2 Proof of Corollary 2

We now instantiate our main Theorem 1 with Theorem 6 to show that lifted codes are robustly testable. For this, we first observe that one can amplify the soundness of the tester given by Theorem 6 to a constant (independent of  $q$  and  $\ell$ ) at the cost of increasing the testing dimension to  $\approx 3\ell$ .

► **Proposition 19.** *Let  $C \subseteq \{\mathbb{F}_q^\ell \rightarrow \mathbb{F}_q\}$  be an affine-invariant linear code, and  $m \geq 3\ell + \log_q 4$ . Then  $C^{\ell \nearrow m}$  is  $(3\ell + \log_q 4, \Omega(1))$ -testable.*

**Proof.** If the  $\ell$ -dimensional test rejects with probability at least  $\frac{1}{2} \cdot \text{dist}(f, C^{\ell \nearrow m})$  then by Part (3) of Proposition 8, the  $(3\ell + \log_q 4)$ -dimensional test also rejects with the same probability and we are done. Otherwise, by Theorem 6, the  $\ell$ -dimensional test rejects with probability at least  $\frac{1}{2} \cdot q^{-2\ell}$ .

Consider the graph  $\mathcal{I}_{\ell, 3\ell + \log_q 4}(0)$  with left hand side being all  $\ell$ -dimensional affine subspaces of  $\mathbb{F}_q^m$  and right hand side being all  $(3\ell + \log_q 4)$ -dimensional affine subspaces of  $\mathbb{F}_q^m$ . Next we apply Propositions 10 and 11 on the graph  $\mathcal{I}_{\ell, 3\ell + \log_q 4}(0)$  with  $A$  being the collection of all  $\ell$ -dimensional affine subspaces on which the  $\ell$ -dimensional test rejects. Noting that the  $(3\ell + \log_q 4)$ -dimensional test will reject on any neighbor of  $A$  we conclude that the  $(3\ell + \log_q 4)$ -dimensional test rejects with probability at least  $1 - \frac{q^{-(2\ell + \log_q 4)}}{q^{-2\ell/2}} = \frac{1}{2}$ . ◀

We now turn to the proof of Corollary 2.

**Proof of Corollary 2.** Suppose first that  $\delta < 2q^{-\ell}$ . In this case by Theorem 6,  $C^{\ell \nearrow m}$  is  $(\ell, \Omega(q^{-2\ell}))$ -testable, and so by Part (2) of Proposition 8,  $C^{\ell \nearrow m}$  is also robustly testable using the  $\ell$ -dimensional test with robustness  $\Omega(q^{-3\ell}) \geq \Omega(\delta^3)$ . By Part (4) of Proposition 8 it follows that the  $(6\ell + \log_q(128/\delta))$ -dimensional test also has robustness  $\Omega(\delta^3)$ .

Next assume that  $\delta \geq 2q^{-\ell}$ . In this case Proposition 4 gives that  $\text{dist}(C^{\ell \nearrow r}) \geq \delta/2$  for any  $\ell \leq r \leq m$ , and so we may apply Proposition 19 and Theorem 1 and conclude that  $C^{\ell \nearrow m}$  is  $(6\ell + \log_q(128/\delta), \Omega(\delta^3))$ -robust. ◀

---

### References

- 1 Sanjeev Arora. *Probabilistic checking of proofs and hardness of approximation problems*. PhD thesis, Princeton University, 1994.
- 2 Sanjeev Arora, Carsten Lund, Rajeev Motwani, Madhu Sudan, and Mario Szegedy. Proof verification and the hardness of approximation problems. *Journal of the ACM*, 45(3):501–555, May 1998.
- 3 Sanjeev Arora and Shmuel Safra. Probabilistic Checking of Proofs: A New Characterization of NP. *Journal of the ACM*, 45(1):70–122, January 1998.
- 4 Sanjeev Arora and Madhu Sudan. Improved Low Degree Testing and its Applications. *Combinatorica*, 23(3):365–426, 2003.
- 5 Eli Ben-Sasson, Oded Goldreich, Prahladh Harsha, Madhu Sudan, and Salil P. Vadhan. Robust PCPs of Proximity, Shorter PCPs, and Applications to Coding. *SIAM J. Comput.*, 36(4):889–974, 2006.
- 6 Eli Ben-Sasson, Prahladh Harsha, and Sofya Raskhodnikova. Some 3CNF Properties Are Hard to Test. *SICOMP: SIAM Journal on Computing*, 35, 2005.
- 7 Eli Ben-Sasson, Ghid Maatouk, Amir Shpilka, and Madhu Sudan. Symmetric LDPC Codes are not Necessarily Locally Testable. In *IEEE Conference on Computational Complexity*, pages 55–65. IEEE Computer Society, 2011. doi:10.1109/CCC.2011.14.
- 8 Eli Ben-Sasson and Madhu Sudan. Robust locally testable codes and products of codes. *Random Struct. Algorithms*, 28(4):387–402, 2006. doi:10.1002/rsa.20120.

- 9 Eli Ben-Sasson and Michael Viderman. Composition of semi-LTCs by two-wise tensor products. *Computational Complexity*, 24(3):601–643, 2015. doi:10.1007/s00037-013-0074-8.
- 10 Manuel Blum, Michael Luby, and Ronitt Rubinfeld. Self-Testing/Correcting with Applications to Numerical Problems. In *STOC*, pages 73–83. ACM, 1990.
- 11 A. E. Brouwer, A. M. Cohen, and A. Neumaier. *Distance-Regular Graphs*. Springer Verlag, 1989.
- 12 Irit Dinur and Elazar Goldenberg. Locally Testing Direct Product in the Low Error Range. In *FOCS*, pages 613–622. IEEE Computer Society, 2008. doi:10.1109/FOCS.2008.26.
- 13 Irit Dinur and Tali Kaufman. High Dimensional Expanders Imply Agreement Expanders. In Chris Umans, editor, *FOCS*, pages 974–985. IEEE Computer Society, 2017. doi:10.1109/FOCS.2017.94.
- 14 Irit Dinur and Inbal Livni-Navon. Exponentially Small Soundness for the Direct Product Z-Test. In Ryan O’Donnell, editor, *Computational Complexity Conference*, volume 79 of *LIPICs*, pages 29:1–29:50. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2017. doi:10.4230/LIPICs.CCC.2017.29.
- 15 Irit Dinur and Omer Reingold. Assignment Testers: Towards a Combinatorial Proof of the PCP Theorem. *SIAM J. Comput*, 36(4):975–1024, 2006. doi:10.1137/S0097539705446962.
- 16 Irit Dinur and David Steurer. Direct Product Testing. In *IEEE Conference on Computational Complexity*, pages 188–196. IEEE Computer Society, 2014. doi:10.1109/CCC.2014.27.
- 17 Peter Gemmell, Richard J. Lipton, Ronitt Rubinfeld, Madhu Sudan, and Avi Wigderson. Self-Testing/Correcting for Polynomials and for Approximate Functions. In Cris Koutsougeras and Jeffrey Scott Vitter, editors, *STOC*, pages 32–42. ACM, 1991. doi:10.1145/103418.103429.
- 18 Goldreich and Safra. A Combinatorial Consistency Lemma with Application to Proving the PCP Theorem. *SICOMP: SIAM Journal on Computing*, 29, 1999.
- 19 Alan Guo, Elad Haramaty, and Madhu Sudan. Robust Testing of Lifted Codes with Applications to Low-Degree Testing. In Venkatesan Guruswami, editor, *FOCS*, pages 825–844. IEEE Computer Society, 2015. doi:10.1109/FOCS.2015.56.
- 20 Alan Guo, Swastik Kopparty, and Madhu Sudan. New affine-invariant codes from lifting. In *ITCS*, pages 529–540, 2013.
- 21 Elad Haramaty, Noga Ron-Zewi, and Madhu Sudan. Absolutely Sound Testing of Lifted Codes. *Theory of Computing*, 11:299–338, 2015. doi:10.4086/toc.2015.v011a012.
- 22 Russell Impagliazzo, Valentine Kabanets, and Avi Wigderson. New Direct-Product Testers and 2-Query PCPs. *SIAM J. Comput*, 41(6):1722–1768, 2012. doi:10.1137/09077299X.
- 23 Tali Kaufman and Madhu Sudan. Algebraic property testing: the role of invariance. In *STOC*, pages 403–412. ACM, 2008. doi:10.1145/1374376.1374434.
- 24 Swastik Kopparty, Or Meir, Noga Ron-Zewi, and Shubhangi Saraf. High-Rate Locally Correctable and Locally Testable Codes with Sub-Polynomial Query Complexity. *Journal of ACM*, 64(2):11:1–11:42, 2017. doi:10.1145/3051093.
- 25 Ran Raz and Shmuel Safra. A Sub-Constant Error-Probability Low-Degree Test, and a Sub-Constant Error-Probability PCP Characterization of NP. In *STOC*, pages 475–484. ACM, 1997. doi:10.1145/258533.258641.
- 26 Ronitt Rubinfeld and Madhu Sudan. Robust characterizations of polynomials with applications to program testing. *SIAM Journal on Computing*, 25(2):252–271, April 1996.
- 27 Michael Viderman. A combination of testability and decodability by tensor products. *Random Struct. Algorithms*, 46(3):572–598, 2015.



# Every Set in $\mathcal{P}$ Is Strongly Testable Under a Suitable Encoding

Irit Dinur<sup>1</sup>

Weizmann Institute, Rehovot, Israel

Oded Goldreich<sup>2</sup>

Weizmann Institute, Rehovot, Israel

Tom Gur<sup>3</sup>

University of Warwick, UK

---

## Abstract

We show that every set in  $\mathcal{P}$  is strongly testable under a suitable encoding. By “strongly testable” we mean having a (proximity oblivious) tester that makes a constant number of queries and rejects with probability that is proportional to the distance of the tested object from the property. By a “suitable encoding” we mean one that is polynomial-time computable and invertible. This result stands in contrast to the known fact that some sets in  $\mathcal{P}$  are extremely hard to test, providing another demonstration of the crucial role of representation in the context of property testing.

The testing result is proved by showing that any set in  $\mathcal{P}$  has a *strong canonical PCP*, where canonical means that (for YES-instances) there exists a single proof that is accepted with probability 1 by the system, whereas all other potential proofs are rejected with probability proportional to their distance from this proof. In fact, we show that  $\mathcal{UP}$  equals the class of sets having strong canonical PCPs (of logarithmic randomness), whereas the class of sets having strong canonical PCPs with polynomial proof length equals “unambiguous- $\mathcal{MA}$ ”. Actually, for the testing result, we use a PCP-of-Proximity version of the foregoing notion and an analogous positive result (i.e., strong canonical PCPPs of logarithmic randomness for any set in  $\mathcal{UP}$ ).

**2012 ACM Subject Classification** Theory of computation → Probabilistic computation

**Keywords and phrases** Probabilistically checkable proofs, property testing

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2019.30

**Related Version** Full technical report hosted on ECCC [9], <https://eccc.weizmann.ac.il/report/2018/050/>.

## 1 Introduction

In the last couple of decades, the area of property testing has attracted much attention (see, e.g., a recent textbook [12]). Loosely speaking, property testing typically refers to super-fast probabilistic algorithms for deciding whether a given object has a predetermined property or is far from any object having this property. Such algorithms, called testers, obtain local views of the object by performing queries; that is, the tested object is modeled as a function and the testers get oracle access to this function (and thus may be expected to work in time that is sub-linear in the size of the object).

---

<sup>1</sup> Supported by ERC-CoG grant 772839. Weizmann Institute of Science.

<sup>2</sup> Supported by the Israel Science Foundation (grants No. 671/13 and 1146/18).

<sup>3</sup> Supported in part by the UC Berkeley Center for Long-Term Cybersecurity.



It is well known that property testing is very sensitive to the representation of the tested objects, far more so than standard studies in complexity theory (cf. [12, Sec. 1.2.5]). For example, while the adjacency matrix and the incident lists representations are equivalent as far as complexity classes such as  $\mathcal{P}$  are concerned, in the case of property testing there is a significant difference between the *adjacency matrix model* (a.k.a. the *dense graph model* [13]) and the *incidence graph model* (a.k.a. the *bounded-degree graph model* [16]).

In this paper we provide another demonstration of the crucial role of representation in the context of property testing. Specifically, in contrast to the known fact that some sets in  $\mathcal{P}$  are extremely hard to test (see, e.g., [15, Theorem 7]),<sup>4</sup> we show that, *under a suitable polynomial-time computable (and invertible) encoding, all sets in  $\mathcal{P}$  are extremely easy to test*, where by “extremely easy to test” we mean having a Proximity Oblivious Tester (POT).

### 1.1 Our main result: a POT for an encoding of any set in $\mathcal{P}$

The standard definition of a property tester refers to randomized oracle machines that are given two parameters as explicit inputs along with oracle access to some string (or function). The two parameters are the *size parameter*, representing the size of the tested object, and a *proximity parameter*, denoted  $\epsilon$ , which determines which objects are considered *far from the property*<sup>5</sup> (according to a fixed metric, typically the relative Hamming distance). Specifically, on input parameters  $n$  and  $\epsilon$ , the test is required to distinguish (with constant probability)  $n$ -bit long strings that have the property from  $n$ -bit long strings that are  $\epsilon$ -far from the property, where  $x \in \{0, 1\}^n$  is  $\epsilon$ -far from  $S$  if for every  $x' \in S \cap \{0, 1\}^n$  it holds that  $\delta(x, x') \stackrel{\text{def}}{=} |\{i \in [n] : x_i \neq x'_i\}|/n$  is greater than  $\epsilon$ . (Otherwise, we say that  $x$  is  $\epsilon$ -close to  $S$ .)

The query complexity of testers is stated as a function of the two explicit parameters,  $n$  and  $\epsilon$ . Two extreme cases are the case of query complexity  $n$ , which can be obtained for any property, and the case that the query complexity depends only on the proximity parameter, which is sometimes considered the yardstick for “easy testability” (see, e.g., [1, 2]). Typically, the query complexity is  $\Omega(1/\epsilon)$ , and so testers of such complexity are extremely efficient. An even more restricted case refers to one in which the tester operates by repeating some constant-query check for  $O(1/\epsilon)$  times, where the celebrated linearity tester of Blum, Luby, and Rubinfeld [7] is an archetypical case. The effect of a single repetition of the constant-query check is captured by the notion of a *proximity oblivious tester* [17].

A Proximity Oblivious Tester (POT) does not obtain a proximity parameter as input, but rather the probability gap with which it distinguishes inputs that have the property from ones that lack the property is allowed to be a function of the distance of the tested input from the property (defined in (1)). Further restricting ourselves to the case of one-sided error testers, we require that the POT always accepts inputs that have the property and rejects objects that lack the property with probability that increases with the distance of the object from the property.<sup>6</sup> For sake of clarity, we recall that the distance of  $x$  from  $S$ , denoted  $\delta_S(x)$ , is defined as follows

$$\delta_S(x) \stackrel{\text{def}}{=} \min_{x' \in \{0,1\}^{|x|} \cap S} \{\delta(x, x')\}, \quad (1)$$

where  $\delta_S(x) = 1$  if  $\{0, 1\}^{|x|} \cap S = \emptyset$ .

<sup>4</sup> This is essentially due to [13, Prop 4.1.1].

<sup>5</sup> As usual in the area, we associate the notion of having a property with the notion of being in the (set of objects that have the) property.

<sup>6</sup> A two-sided error version was also studied (see [18]), but the one-sided error version that we consider here is much better known.

► **Definition 1.1** (Proximity Oblivious Testers).<sup>7</sup> Let  $\varrho: (0, 1] \rightarrow (0, 1]$  be monotonically non-decreasing.<sup>8</sup> A proximity oblivious tester (POT) with detection probability function  $\varrho$  for  $S$  is a probabilistic oracle machine, denoted  $T$ , that makes a constant number of queries and satisfies the following two conditions.

1.  $T$  always accepts inputs in  $S$ : For every  $n \in \mathbb{N}$  and every  $w \in S \cap \{0, 1\}^n$ , it holds that  $\Pr[T^w(n) = 1] = 1$ .
2.  $T$  rejects inputs that are not in  $S$  with probability that increases as a function of their distance from  $S$ : For every  $n \in \mathbb{N}$  and every  $w \in \{0, 1\}^n \setminus S$ , it holds that  $\Pr[T^w(n) = 0] \geq \varrho(\delta_S(w))$ .

The case that  $\varrho$  is linear is of special interest; in this case the rejection probability is proportional to the distance of the input from the set  $S$ .

Our main result asserts the existence of a POT for some encoding of any set in  $\mathcal{P}$ . Starting with some natural representation of a set  $S \subseteq \{0, 1\}^*$ , we consider a representation obtained by applying an invertible encoding  $E: \{0, 1\}^* \rightarrow \{0, 1\}^*$  (i.e., we require that  $E$  is one-to-one). Furthermore, we consider the natural case in which this encoding is polynomial-time computable and invertible. For example, we may consider an encoding such as  $E(x_1 \cdots x_n) = x_1 x_1 \cdots x_n x_n$  or encodings that map graphs in the adjacency matrix representation to the incidence list representation.

► **Theorem 1.2** (a POT for a suitable encoding of any set in  $\mathcal{P}$ ). *For any  $S \in \mathcal{P}$  there exist polynomial-time encoding and decoding algorithms  $E$  and  $D = E^{-1}$  such that the set  $S' \stackrel{\text{def}}{=} \{E(x) : x \in S\}$  has a proximity oblivious tester of linear detection probability. Furthermore,  $|E(x)| = |E(1^{|x|})|$  for every  $x$ , the encoding  $E$  has constant relative distance,<sup>9</sup> and the POT runs in polylogarithmic time and has logarithmic randomness complexity.*

Recall that POTs were defined as having constant query complexity, and note that the added conditions regarding  $E$  (i.e., being “length regular” and having constant relative distance) only make the result potentially more appealing. Theorem 1.2 cannot be significantly extended, since the existence of a polynomial-time tester (of arbitrary query complexity) for  $\{E(x) : x \in S\}$  such that  $E$  is polynomial-time computable implies that  $S \in \mathcal{BPP}$  (and  $S \in \mathcal{P}$  follows if the tester has logarithmic randomness complexity).<sup>10</sup>

## 1.2 The way to our main result: strong canonical PCPPs

Theorem 1.2 is proved by using an encoding that maps the input  $x$  to a pair of the form  $(C(x)^{t(|x|)}, \Pi(x))$ , where  $C$  is an error-correcting code,  $\Pi(x)$  is a PCP proof that  $C(x) \in \{C(z) : z \in S\}$ , and  $t(|x|) \approx |\Pi(x)|/|C(x)|$  (so that the two parts of the pair have approximately the same length). This idea, which can be traced back to [19], works only when the PCP system is of a certain type, as discussed next.

<sup>7</sup> Unlike in [17], which considered POTs of arbitrary query complexity, here we mandate that a POT has constant query complexity. This choice is justified by the fact that our result establishes the existence of such POTs. Ditto regarding our choice to consider one-sided error only.

<sup>8</sup> The postulate that  $\varrho$  is monotonically non-decreasing means that *any input that is  $\epsilon$ -far from  $S$  is rejected with probability at least  $\varrho(\epsilon)$* ; that is, if  $\delta_S(f) \geq \epsilon$  (and not only if  $\delta_S(f) = \epsilon$ ), then  $f$  is rejected with probability at least  $\varrho(\epsilon)$ . This postulate is natural (and it can be enforced in general by redefining  $\varrho(\epsilon) \leftarrow \inf_{\delta \geq \epsilon} \{\varrho(\delta)\}$ ).

<sup>9</sup> That is, for every  $x \neq y$ , the encodings  $E(x)$  and  $E(y)$  differ on a constant fraction of the coordinates.

<sup>10</sup> The decision procedure maps  $x$  to  $E(x)$  and invokes the tester with proximity parameter  $1/2|E(x)|$ . In case  $E$  has relative distance  $\delta$ , invoking the tester with proximity parameter  $\delta/2$  will do.

First, we need a PCP of Proximity (a.k.a. assignment testers), rather than a PCP. The difference is that a PCP of Proximity does not get an explicit (main) input, but rather oracle access to both the main input and the alleged proof. Indeed, PCPs of Proximity can be viewed as a “property testing” variant of PCPs (or “PCP-aided variants” of property testers). Second, valid assertions in these PCPs of Proximity must have *unique* valid proofs, otherwise the mapping  $x \mapsto \Pi(x)$  is not even well-defined. Furthermore, the PCP of Proximity should reject (with constant probability) not only inputs (that encode) strings far from  $S$ , but also proof-parts that are far from the corresponding (unique) valid proof. Last, to get a POT rather than a tester that works only for constant values of the proximity parameter (i.e., constant  $\epsilon > 0$ ), also inputs and alleged proofs that are close to being valid should be rejected with probability that is related to their distance from a valid object. A PCP of Proximity that satisfies all of these conditions is called *strongly canonical*.<sup>11</sup>

The foregoing aspects were dealt with in [19], but only for the special case of  $S = \{0, 1\}^*$ , where the issue was to test that the input-part is a valid codeword (with respect to code  $C$ ). Using a linear code  $C$ , this was reduced to the special case in which the set (for which the PCP of Proximity is designed) is a linear space, but even this case was not handled in full generality in [19]. Subsequent work [21, 14, 20] culminated in providing strongly canonical PCPs of Proximity for any linear space, but left open the problem of providing strongly canonical PCPs of Proximity for any set in  $\mathcal{P}$ , let alone  $\mathcal{UP}$ .

Recall that the class  $\mathcal{UP}$  is defined as the subset of  $\mathcal{NP}$  in which each YES-instance has a unique valid proof. In this paper, we show that *every set in  $\mathcal{UP}$  has a strong canonical PCP of Proximity*. Furthermore, we provide a polynomial-time transformation of NP-witnesses (with respect to the original NP-witness relation of the set) to valid proofs (for the resulting PCP of Proximity).

We seize the opportunity to study the simpler case of strong canonical PCPs. Loosely speaking, a **strong canonical PCP** for a set  $S$  is a PCP system in which each  $x \in S$  has a unique valid proof  $\Pi(x)$  that is accepted with probability 1, whereas each other alleged proof is rejected with probability that is related to its distance from  $\Pi(x)$ . We show that:

1. *Every set in  $\mathcal{UP}$  has a strong canonical PCP of logarithmic randomness, and only sets in  $\mathcal{UP}$  have such a PCP* (see Theorem 3.1).
2. *Similarly, the class of sets having (sufficiently)<sup>12</sup> strong canonical PCPs with polynomial proof length equals “unambiguous-MA”* (see Theorem 3.4).

All our constructions are obtained in two steps. First, we show that sets in the relevant class have PCP systems in which each string in the set has a unique valid proof (that is accepted with probability 1). Specifically, we show that the only proofs that are accepted with probability 1 by these PCP systems are the images of the standard transformation of NP-witnesses to PCP-oracles. Next, we observe that these PCP systems can be made strongly canonical by a suitable padding of the proofs. Specifically, the padding is determined such that the ratio of the length of the original proof over the length of the padded proof equals the lower bound on the rejection probability of invalid proofs (under the original PCP). Indeed, this simple observation reduces the construction of strong canonical PCPs to the construction of PCPs that have unique valid proofs.

<sup>11</sup> See Definition 2.2, which requires that the rejection probability of the oracle pair  $(x, \pi)$  be related to the maximum, over all  $x' \in \{0, 1\}^{|x|} \cap S$ , of  $\delta(x, x')$  and  $\delta(\pi, \Pi(x'))$ .

<sup>12</sup> Here we require that any alleged proof is rejected with probability that is *polynomially related* to its distance from  $\Pi(x)$  (i.e.,  $\varrho(\delta) = \text{poly}(\delta)$ ).

Focusing on the construction of PCPs that have unique valid proofs (for sets in  $\mathcal{UP}$ ), we note that the original PCP construction of Arora et al. [4, 3] will not do. Still, it is possible that this construction can be modified and augmented so that it has unique valid proofs (or even becomes a strong canonical PCP). Such an augmentation was indeed performed by Goldreich and Sudan [19], alas only for the special case of linear spaces, and the route taken there was quite tedious. Hence, we preferred to work with the gap amplification construction of Dinur [8], which is more transparent. Starting with a trivial weak-PCP that has unique valid proofs, we observe that the gap amplification operation is a parsimonious reduction, and so we are done.

### 1.3 Organization

In Section 2 we recall the definitions of strong canonical PCPs and PCPPs, starting with the basic PCP model. In Section 3 we characterize the classes of sets having strong canonical PCPs of certain types (see Theorems 3.1 and 3.4), and obtain analogous PCPP systems (see Theorem 3.5). The latter PCPP systems will be used in Section 4 towards establishing Theorem 1.2. We conclude by spelling out some directions for further research (see Section 5).

## 2 Definitions of strong canonical PCPs and PCPPs

In this section, we recall the definitions of strong canonical PCPs and PCPPs. Essentially, we follow the definitional approach presented in [19, Sec. 5.3] (while correcting an error in one of the actual definitions [19, Def. 5.7]).

### 2.1 Preliminaries: The PCP model

We start by recalling the basic definition of Probabilistically Checkable Proofs (PCPs): These are randomized verification procedures that are given an explicit input and oracle access to an alleged proof  $\pi$ , and are aimed to verify the membership of the (main) input in a predetermined set by making few (random) queries to the proof (see, e.g., [11, Sec. 9.3]). Specifically, for a predetermined set  $S \subseteq \{0, 1\}^*$ , on input  $x$  and oracle access to an alleged proof  $\pi$ , a PCP verifier  $V$  reads  $x$ , makes a constant number of random queries to the proof  $\pi$ , and satisfies the following conditions.

**Completeness:** If  $x \in S$ , then there exists a valid proof  $\pi$  such that  $V$  always accepts  $x$  when given oracle access to  $\pi$ ; that is,  $\Pr[V^\pi(x) = 1] = 1$ .

**Soundness:** If  $x \notin S$ , then for every string  $\pi$ , with probability at least  $1/2$  the verifier  $V$  rejects  $x$  when given oracle access to  $\pi$ ; that is,  $\Pr[V^\pi(x) = 0] \geq 1/2$ .<sup>13</sup>

Indeed, a string  $\pi$  that makes  $V$  always accept  $x$  (i.e., that satisfies  $\Pr[V^\pi(x) = 1] = 1$ ) is called a **valid proof for  $x$** ; the soundness condition implies that valid proofs exist only for members of  $S$ , and the completeness condition asserts that each member of  $S$  has a valid proof. We stress that it is not necessarily the case that the valid proofs are unique; that is, the same  $x \in S$  may have several valid proofs (with respect to a fixed verifier).

<sup>13</sup> Actually, the constant  $1/2$  can be replaced by any other constant in  $(0, 1)$ .

### Weak-PCPs

We shall also refer to the notion of a *weak-PCP*, which is defined as above with the crucial exception that its soundness condition is extremely weak. Specifically, this weak soundness condition only requires that for every  $x \notin S$  and  $\pi$ , with positive probability, the verifier rejects  $x$  when given oracle access to  $\pi$  (i.e.,  $\Pr[V^\pi(x)=0] > 0$ ). Indeed, an oracle machine that on input a 3CNF and oracle access to a truth assignment to its variables checks the values assigned to the variables of a uniformly selected clause constitutes such a trivial weak-PCP. (Recall that Dinur's construction [8], which we shall use, gradually transforms such a weak-PCP into a full fledged PCP.)

## 2.2 Strong canonical PCPs

We focus on the special case of PCP verifiers, for a set  $S$ , with respect to which each  $x \in S$  has a *unique* valid proof, and call such verifiers **canonical**. Furthermore, we are interested in the case that invalid proofs are not merely rejected with positive probability, but are rather rejected with probability that is related to their distance from the (unique) valid proof. We shall call such verifiers **strongly canonical**, and quantify their strength by a function  $\varrho$  that relates their rejection probability to the latter distance. Details follow.

We denote the empty string by  $\lambda$ . For two strings  $w, w' \in \{0, 1\}^m$ , we let  $\delta(w, w')$  denote the relative Hamming distance between  $w$  and  $w'$ ; that is,  $\delta(w, w') = |\{i \in [m] : w_i \neq w'_i\}|/m$ . For sake of convenience, we define  $\delta(w, w') = 1$  if  $w$  and  $w'$  have different lengths (e.g., the distance between a non-empty string and the empty string is 1).

► **Definition 2.1** (strong canonical PCPs). For a set  $S \subseteq \{0, 1\}^*$ , a monotonically non-decreasing function  $\varrho: [0, 1] \rightarrow [0, 1]$  such that  $\varrho(\alpha) = 0$  if and only if  $\alpha = 0$ , and an oracle machine  $V$ , we say that  $V$  is a  $\varrho$ -strong canonical PCP for  $S$  if  $V$  makes a constant number of queries to the oracle and there exist functions  $\ell: \mathbb{N} \rightarrow \mathbb{N}$  and  $\Pi: \{0, 1\}^* \rightarrow \{0, 1\}^*$  such that the following conditions hold.

- **Canonical Completeness:** For every  $x \in S$ , it holds that  $\Pi(x) \in \{0, 1\}^{\ell(|x|)}$ , and the verifier always accepts  $x$  when given oracle access to  $\Pi(x)$ ; that is,  $\Pr[V^{\Pi(x)}(x)=1] = 1$ .
- **Strong Canonical Soundness:** For every  $x \in \{0, 1\}^*$  and  $\pi \in \{0, 1\}^*$ , the verifier rejects  $x$  when given access to the oracle  $\pi$  with probability at least  $\varrho(\delta(\pi, \Pi(x)))$ , where  $\Pi(x) \stackrel{\text{def}}{=} \lambda$  if  $x \notin S$  (and in this case  $\delta(\pi, \Pi(x)) = 1$ ); that is,  $\Pr[V^\pi(x)=0] \geq \varrho(\delta(\pi, \Pi(x)))$ .

The function  $\varrho$  is called  $V$ 's detection probability function, and  $\ell$  is called its **proof complexity**. We say that  $V$  is a **strong canonical PCP** for  $S$  if, for some  $\varrho$  as above,  $V$  is a  $\varrho$ -strong canonical PCP for  $S$ .

Indeed, the foregoing conditions assert that  $\Pi(x)$  is the unique valid proof for  $x \in S$ , and that the verifier is strongly canonical with strength  $\varrho$ . Note that strong canonical soundness implies (standard) soundness by the convention that  $\varrho(\delta(\pi, \Pi(x))) = \varrho(1) = \Omega(1)$  for  $x \notin S$ . More generally, recall that  $\delta(\pi, \Pi(x)) = 1$  if  $|\pi| \neq |\Pi(x)|$ . The case that  $\varrho$  is linear is of special interest; in this case invalid proofs are rejected with probability that is proportional to their distance from the valid proof.

## 2.3 Adaptation to the model of PCP of Proximity

Probabilistically checkable proofs of proximity (PCPs of Proximity, abbreviated PCPPs and a.k.a. assignment testers) are proof systems in which the verifier has oracle access to both its main input and an alleged proof, and is required to decide whether the main input is in



some predetermined set or is far from any string that is in this set (cf. [5, 10]). We call such a PCPP system **strong** if it rejects every NO-instance with probability that is related to the distance of the instance from the predetermined set. For simplicity, when we say a PCPP system, we mean a strong one.

Analogously to the case of PCPs, we consider **strong canonical PCPs of Proximity**<sup>14</sup> (henceforth scPCPs of Proximity), which are PCPs of Proximity in which every statement has a unique valid proof such that a statement–proof pair is rejected with probability that is related to its distance from a true statement and its corresponding unique valid proof. The actual definition builds on Definition 2.1, while adapting it to the proofs of proximity model.

► **Definition 2.2** (strong canonical PCPs of Proximity). For a set  $S$ , a function  $\varrho$  as in Definition 2.1, and an oracle machine  $V$  that accesses two oracles, we say that  $V$  is a  $\varrho$ -strong canonical PCP of Proximity for  $S$  if  $V$  makes a constant number of queries to each of its oracles and there exist functions  $\ell: \mathbb{N} \rightarrow \mathbb{N}$  and  $\Pi: \{0, 1\}^* \rightarrow \{0, 1\}^*$  such that the following conditions hold.

- **Canonical Completeness:** For every  $x \in S$ , it holds that  $\Pi(x) \in \{0, 1\}^{\ell(|x|)}$  and the verifier always accepts the pair of oracles  $(x, \Pi(x))$ ; that is,  $\Pr[V^{x, \Pi(x)}(1^{|x|}) = 1] = 1$ .
- **Strong Canonical Soundness:** For every  $x \in \{0, 1\}^*$  and  $\pi \in \{0, 1\}^*$ , the verifier rejects the pair of oracles  $(x, \pi)$  with probability at least  $\varrho(\delta_\Pi(x, \pi))$ , where<sup>15</sup>

$$\delta_\Pi(x, \pi) \stackrel{\text{def}}{=} \min_{x' \in \{0, 1\}^{|x|}} \{\max(\delta(x, x'), \delta(\pi, \Pi(x')))\}; \quad (2)$$

that is,  $\Pr[V^{x, \pi}(1^{|x|}) = 0] \geq \varrho(\delta_\Pi(x, \pi))$ .

The function  $\varrho$  is called  $V$ 's detection probability function, and  $\ell$  is called its proof complexity. We say that  $V$  is a **strong canonical PCPP** for  $S$  if, for some  $\varrho$  as above,  $V$  is a  $\varrho$ -strong canonical PCPP for  $S$ .

We stress that the rejection probability depends on the distance of the oracle-pair  $(x, \pi)$  from a valid pair consisting of  $x' \in S \cap \{0, 1\}^{|x|}$  and the corresponding valid proof  $\Pi(x')$ , where the distance between pairs is defined as the maximum of the distance between the corresponding elements.<sup>16</sup> This represents the fact that we wish to reject with probability that not only depends on the distance of the input  $x$  to a string  $x' \in S$ , but also depends on the distance of the alleged proof  $\pi$  to the corresponding valid proof  $\Pi(x')$ . Indeed, proximity oblivious testers (POTs) can be viewed as strong canonical PCPs of Proximity with proof complexity zero.

### 3 On the existence of strong canonical PCPs and PCPPs

Our first result is a characterization of the class of sets having strong canonical PCPs of *logarithmic randomness*. It turns out that this class equals  $\mathcal{UP}$ . Recall that the class  $\mathcal{UP}$  is defined as the subset of  $\mathcal{NP}$  in which each YES-instance has a unique valid proof; that is,  $S \in \mathcal{UP}$  if there exists a polynomially-bounded relation  $R$  that is recognizable in polynomial-time such that for every  $x \in S$  there exists a unique  $w \in R(x) = \{y : (x, y) \in R\}$  whereas  $R(x) = \emptyset$  if  $x \notin S$ .

<sup>14</sup> Alternatively, we use the term *strongly canonical*.

<sup>15</sup> Recall that  $\Pi(x') \stackrel{\text{def}}{=} \lambda$  if  $x' \notin S$ , and in this case  $\delta(\pi, \Pi(x')) = 1$ .

<sup>16</sup> That is, we effectively define  $\delta(\langle x, y \rangle, \langle x', y' \rangle)$  as  $\max(\delta(x, x'), \delta(y, y'))$ . Taking the sum of the latter distances (or their average) would have been as good, since  $\frac{\alpha + \beta}{2} \leq \max(\alpha, \beta) \leq \alpha + \beta$ .

► **Theorem 3.1** (*UP and strong canonical PCPs*). *The set  $S$  has a strong canonical PCP of logarithmic randomness if and only if  $S \in \mathcal{UP}$ . Furthermore, the resulting PCP is  $\varrho$ -strong for  $\varrho(\alpha) = \alpha/4$  and there exists a polynomial-time transformation of NP-witnesses for  $S \in \mathcal{UP}$  to valid proofs for the resulting PCP.*

**Proof.** The necessary condition is quite straightforward: Let  $V$  be a strong canonical PCP of logarithmic randomness for  $S$ , and assume for simplicity that its proof complexity is polynomial. Now, define  $R = \{(x, \pi) : \Pr[V^\pi(x) = 1] = 1\}$ , and observe that membership in  $R$  can be decided in polynomial-time by trying all possible random choices of  $V$ . Hence,  $S = \{x : R(x) \neq \emptyset\}$  is in  $\mathcal{NP}$ , and the hypothesis that the valid proofs (with respect to  $V$ ) are unique implies that  $S \in \mathcal{UP}$ . In the general case (i.e., when the proof length may be super-polynomial), one may consider the “effective proofs” (i.e., the values of  $\pi$  at locations that are read by  $V$  on some random choices). That is, in this case, we define  $R = \{(x, (I(x), \pi_{I(x)})) : \Pr[V^\pi(x) = 1] = 1\}$ , where  $I(x)$  is the set of locations that are in the “effective proof” (i.e., locations that  $V(x)$  probes with positive probability).

Note that the foregoing argument holds also for very weak PCP systems, provided that they have logarithmic randomness complexity and unique valid proofs. That is, we only used the hypothesis that for every  $x \in S$ , there exists a unique  $\pi$  such that  $p_x(\pi) \stackrel{\text{def}}{=} \Pr[V^\pi(x) = 1] = 1$ , and capitalized on the fact that it is feasible to compute  $p_x(\pi)$  exactly.

Turning to the opposite direction, we show that each  $S \in \mathcal{UP}$  has a strong canonical PCP of logarithmic randomness by presenting such a PCP for **USAT** and recalling that each set in  $\mathcal{UP}$  is reducible to **USAT** via a parsimonious reduction (see, e.g., [11, Ex 2.29]). Recall that **USAT** is the promise problem in which **YES**-instances are 3CNF formulas with a *unique* satisfying assignment and **NO**-instances are formulas with no satisfying assignments. (Actually, we need to define PCPs for promise problems and state, as well as prove, Proposition 3.2 in this more general setting, but we avoid doing so while commenting that the extension is straightforward.)<sup>17</sup>

The key observation is that it suffices to show a PCP for  $S$  in which each  $x \in S$  has a unique valid proof. This is the case because such PCPs can be transformed into strong canonical ones, as stated next.

► **Proposition 3.2** (*deriving strong canonical PCPs from PCPs with unique valid proofs*). *Let  $V$  be a PCP system of logarithmic randomness complexity for  $S$ , and suppose that for every  $x \in S$  there exists a unique  $\pi$  such that  $\Pr[V^\pi(x) = 1] = 1$ . Then, there exist a strong canonical PCP of logarithmic randomness for  $S$  and a polynomial-time transformation of valid proof with respect to  $V$  to valid proofs for the resulting PCP. Furthermore, the resulting PCP is  $\varrho$ -strong for  $\varrho(\alpha) = \alpha/4$  and its proof complexity is  $2^r \cdot \ell$ , where  $r$  and  $\ell$  are the randomness and proof complexity of  $V$ .*

**Proof.** Again, we may assume, without loss of generality, that  $V$  has polynomial proof complexity, since we can efficiently determine all relevant locations (i.e., those queried under any choice of randomness) without making any queries.

Letting  $\Pi$  be the function mapping instances to their canonical proofs, as in Definition 2.1, we define  $\Pi'(x) = \mathbf{1}_{(2^{r(|x|)} - 1) \cdot \ell(|x|)} \Pi(x) \in \{0, 1\}^{2^{r(|x|)} \cdot \ell(|x|)}$  if  $\Pi(x) \neq \lambda$  and  $\Pi'(x) = \lambda$  otherwise. Note that, for every  $x \in S$  and  $\pi \in \{0, 1\}^{\ell(|x|)}$ , it holds that

$$\delta(\mathbf{1}_{(2^{r(|x|)} - 1) \cdot \ell(|x|)} \pi, \Pi'(x)) \leq 2^{-r(|x|)},$$

<sup>17</sup> Specifically, the canonical soundness condition has to be satisfied only for inputs that satisfy the promise, whereas the canonical completeness condition is stated for the **YES**-instances only.



which means that invalid proofs for  $x \in S$  are extremely close to valid proofs, and so it suffices to reject them with tiny probability. This suggests the following verifier, which on input  $x \in \{0, 1\}^n$  and access to oracle  $\pi' \in \{0, 1\}^{2^{r(n)} \cdot \ell(n)}$ , selects uniformly at random one of the following two tests and performs it.

1. The verifier selects at random  $i \in [(2^{r(n)} - 1) \cdot \ell(n)]$ , and accepts if and only if the  $i^{\text{th}}$  bit of  $\pi'$  equals 1.
2. The verifier invokes  $V$  on input  $x$ , while providing it with oracle access to the  $\ell(n)$ -bit long suffix of  $\pi'$ , and outputs the verdict of  $V$ .

Turning to the analysis, we first note that if  $x \notin S$ , then (by virtue of  $V$ ) the resulting verifier rejects  $x$  with probability at least  $\frac{1}{2} \cdot \frac{1}{2}$ , regardless of the identity of the oracle  $\pi'$ . Hence, from this point on we assume  $x \in S$ , and let  $\Pi(x)$  denote the unique valid proof with respect to  $V$ .

Now, let  $\pi' \equiv (w, \pi) \in \{0, 1\}^{2^{r(n)} \cdot \ell(n)}$  such that  $|\pi| = \ell(n)$ . Observe that, on input  $x$  and access to  $\pi'$ , the new verifier rejects with probability that is lower-bounded by (half) the fraction of 0's in  $w$ , since with probability  $1/2$  this verifier test whether the  $(2^{r(n)} - 1) \cdot \ell(n)$ -bit long prefix equals the all-1 string. Next, recall that, for any  $\pi \in \{0, 1\}^{\ell(n)}$ , it holds that  $\pi'' = 1^{(2^{r(n)} - 1) \cdot \ell(n)} \pi$  is  $2^{-r(|x|)}$ -close to  $\Pi'(x) = 1^{(2^{r(n)} - 1) \cdot \ell(n)} \Pi(x)$ , since  $\delta(\pi, \Pi(x)) \leq 1$ . Hence, rejecting  $\pi''$  with probability at least  $\frac{1}{2} \cdot 2^{-r(n)}$  suffices when  $\pi \neq \Pi(x)$ . It follows that a generic  $\pi' \equiv (w, \pi) \neq \Pi'(x) \neq \lambda$  is rejected with probability

$$\begin{aligned} \frac{1}{2} \cdot \delta(w, 1^{(2^{r(n)} - 1) \cdot \ell(n)}) + \frac{1}{2} \cdot 2^{-r(n)} &\geq \frac{1}{2} \cdot \delta(w, 1^{(2^{r(n)} - 1) \cdot \ell(n)}) + \frac{1}{2} \cdot 2^{-r(n)} \cdot \delta(\pi, \Pi(x)) \\ &\geq \frac{1}{2} \cdot \delta(\pi', \pi'') + \frac{1}{2} \cdot \delta(\pi'', \Pi'(x)), \end{aligned}$$

where the second inequality uses  $\delta(w, 1^{(2^{r(n)} - 1) \cdot \ell(n)}) \geq \delta(w\pi, 1^{(2^{r(n)} - 1) \cdot \ell(n)} \pi) = \delta(\pi', \pi'')$  and

$$\begin{aligned} \delta(\pi, \Pi(x)) &= 2^{r(|x|)} \cdot \delta(1^{(2^{r(n)} - 1) \cdot \ell(n)} \pi, 1^{(2^{r(n)} - 1) \cdot \ell(n)} \Pi(x)) \\ &= 2^{r(|x|)} \cdot \delta(\pi'', \Pi'(x)). \end{aligned}$$

Using  $\delta(\pi', \pi'') + \delta(\pi'', \Pi'(x)) \geq \delta(\pi', \Pi'(x))$ , it follows that, on input  $x$  and access to  $\pi'$ , the new verifier rejects with probability at least  $\delta(\pi', \Pi'(x))/2$ , which means that it constitutes a strong canonical PCP for  $S$ . Indeed, the PCP is  $\varrho$ -strong for  $\varrho(\alpha) = \alpha/4$ .<sup>18</sup> ◀

In light of Proposition 3.2, it suffices to show a PCP of logarithmic randomness and unique valid proofs for USAT. This PCP is constructed by merely following the construction of Dinur [8], while noting that her gap amplification transformation is a parsimonious reduction.

► **Proposition 3.3** (PCPs with unique valid proofs for USAT). *There exists a PCP system of logarithmic randomness for USAT such that for every satisfiable formula there exists a unique valid proof with respect to this system. Furthermore, there exists a polynomial-time transformation of satisfying assignments for the input formula to valid proofs for the resulting PCP.*

**Proof.** Let  $\psi$  be an  $m$ -clause 3CNF formula over  $n$  variables, promised to have at most one satisfying assignment. Let  $V_0$  be the trivial weak-PCP system with soundness  $1/m$ , in which the oracle is allegedly the unique satisfying assignment of  $\psi$ , and the verifier checks that

<sup>18</sup>The factor of  $1/4$  is due to the case that  $x \notin S$ , which is rejected with probability at least  $1/4$ .

this assignment satisfies a random clause of  $\psi$ . By construction,  $V_0$  has unique valid proofs. Applying the gap amplification transformation of Dinur [8] to  $V_0$ , we obtain a PCP system  $V$  of logarithmic randomness for USAT.

As stated above, the crucial point is showing that the aforementioned transformation is a parsimonious reduction. The argument is detailed in our technical report [9, Appendix A]; it consists of showing that gap amplification is a one-to-one transformation, and that the only valid proofs with respect to the resulting proof system are those in the range of the transformation. These facts are demonstrated by closely inspecting each of the four steps in the gap amplification procedure: degree reduction, “expanderization”, powering, and alphabet reduction. We show that each of these steps satisfies the two aforementioned properties, where in the analysis of the alphabet reduction step we assume that it is performed by composition with a PCPP that has unique valid proofs. Such a PCPP is immediately implied by the Hadamard code (alternatively, by the long code); see details in [9, Appendix A.4]. ◀

Combining Proposition 3.2 and 3.3, the theorem follows. ◀

### A detour: A variant of Theorem 3.1

Our next result is a characterization of the class of sets having strong canonical PCPs of *polynomial proof length*. It turns out that this class equals “unambiguous- $\mathcal{MA}$ ” (denoted  $\mathcal{UM\mathcal{A}}$ , and defined next). Recall that the class  $\mathcal{MA}$  consists of all sets having a non-interactive probabilistic proof system; that is,  $S \in \mathcal{MA}$  if there exists a polynomially-bounded relation  $R$  that is recognizable in  $\text{co}\mathcal{RP}$  such that  $S = \{x : \exists w (x, w) \in R\}$ .<sup>19</sup> We define  $\mathcal{UM\mathcal{A}}$  as the subset of  $\mathcal{MA}$  in which the non-interactive proof system has unique valid proofs; that is,  $S \in \mathcal{UM\mathcal{A}}$  if there exists a polynomially-bounded relation  $R$  that is recognizable in  $\text{co}\mathcal{RP}$  such that for every  $x \in S$  there exists a unique  $w \in R(x) = \{y : (x, y) \in R\}$ , whereas  $R(x) = \emptyset$  if  $x \notin S$ . (Note that in this case  $|R(x)| \leq 1$  for every  $x$ .)

► **Theorem 3.4** ( *$\mathcal{UM\mathcal{A}}$  and strong canonical PCPs*). *The set  $S$  has a poly-strong canonical PCP of polynomial proof complexity if and only if  $S \in \mathcal{UM\mathcal{A}}$ .*

Above, recall that by poly-strong we mean  $\varrho$ -strong with respect to  $\rho(\delta) = \delta^c$ . We stress that, unlike in Theorem 3.1, here we do not know whether a  $\varrho$ -strong canonical PCP with arbitrary  $\varrho$  for  $S$  implies that  $S \in \mathcal{UM\mathcal{A}}$ . The point is that invalid proofs of length  $\ell$  are only guaranteed to be rejected with probability at least  $\varrho(1/\ell)$ , which may be negligible. On the other hand, the existence of poly-strong canonical PCP (of polynomial-length) for  $S$ , implies  $S \in \mathcal{UM\mathcal{A}}$ , which in turn is shown to imply that  $S$  has a  $\varrho$ -strong canonical PCP (of polynomial-length) with a linear  $\varrho$  (i.e.,  $\varrho(\alpha) = \Omega(\alpha)$ ).

**Proof.** We follow the outline of the proof of Theorem 3.1, while introducing several relevant modifications. For example, in the proof of the necessary condition we can no longer assume that the PCP has logarithmic randomness; instead we directly use the hypothesis that the PCP has polynomial proof complexity, and derive a verification procedure that places the set in  $\mathcal{UM\mathcal{A}}$  (rather than in  $\mathcal{UP}$ ). Furthermore, using the hypothesis that the PCP is poly-strong, we infer that invalid proofs are rejected with noticeable probability (i.e., probability at least  $\varrho(1/\ell) = \text{poly}(1/\ell)$ ). This fact allows for the rejection of invalid proofs by invoking the PCP

<sup>19</sup>This perfect completeness version of  $\mathcal{MA}$  equals the non-perfect one in which  $R$  is only required to be recognizable in  $\mathcal{BPP}$  (see [11, Ex. 6.12 (2)]).

verifier polynomially many times. Specifically, let  $V$  be a  $\varrho$ -strong canonical PCP of proof complexity  $\ell = \text{poly}$  for  $S$ , and define  $R = \{(x, \pi) : \Pr[V^\pi(x) = 1] = 1\}$ . Then,  $R \in \text{coRP}$ , by letting the decision procedure emulate  $O(1/\varrho(1/\ell(|x|))) = \text{poly}(|x|)$  executions of  $V^\pi(x)$ , and accept if and only if all executions accepted. Hence,  $S = \{x : R(x) \neq \emptyset\}$  is in  $\mathcal{MA}$ , and the hypothesis that the valid proofs (with respect to  $V$ ) are unique implies that  $S \in \mathcal{UMMA}$ .

Turning to the opposite direction, we show that each  $S \in \mathcal{UMMA}$  has a poly-strong canonical PCP of polynomial proof length, by using a randomized reduction of  $S$  to  $\text{USAT}$  (or rather to a promise problem in the corresponding class  $\mathcal{UP}$ ). Let  $R$  be the binary relation guaranteed by the definition of  $\mathcal{UMMA}$ , and suppose, without loss of generality, that  $R \subseteq \cup_{n \in \mathbb{N}} (\{0, 1\}^n \times \{0, 1\}^{p(n)})$  for some polynomial  $p$ . Let  $p'$  be a polynomial that upper bounds the randomness complexity of the decision procedure for  $R$ , and let  $D'$  denote the residual decision predicate of that procedure; that is,  $D'_{r_i}(x, w)$  denotes the verdict on input  $(x, w)$  when using randomness  $r \in \{0, 1\}^{p'(n+p(n))}$ . Recall that for  $(x, w) \notin R$ , it holds that  $\Pr_r[D'_{r_i}(x, w) = 1] \leq 1/2$ , and it follows that, for every  $x$  and  $w \notin R(x) = \{y : (x, y) \in R\}$ ,

$$\Pr_{r_1, \dots, r_m \in \{0, 1\}^{p'(n+p(n))}} [\forall i \in [m] D'_{r_i}(x, w) = 1] \leq 2^{-m}.$$

Note that if we pick  $m = p(n) + 2$ , then an application of a union bound implies that, for every  $x \in \{0, 1\}^n$ , it holds

$$\Pr_{r_1, \dots, r_m \in \{0, 1\}^{p'(n+p(n))}} [\exists w \notin R(x) \forall i \in [m] D'_{r_i}(x, w) = 1] \leq 1/4.$$

Now, consider the randomized mapping of  $x \in \{0, 1\}^n$  to  $(x, r_1, \dots, r_m)$ , denoted  $\Psi$ , where  $m = p(n) + 2$  and the  $r_i$ 's are selected uniformly and independently in  $\{0, 1\}^{p'(n+p(n))}$ . Recall that  $|R(x)| \leq 1$  for any  $x$ . Now, let  $P$  (standing for promise) denote the set of tuples  $(x, r_1, \dots, r_m)$  for which  $\forall i \in [m] D'_{r_i}(x, w) = 1$  holds only for  $w \in R(x)$ , and  $S'$  denote the set of tuples  $(x, r_1, \dots, r_m)$  with  $x \in S$ . Then, it holds that  $\Pr[\Psi(x) \in P] \geq 3/4$  and  $\Pr[\Psi(x) \in S' \Leftrightarrow x \in S] = 1$  for each  $x$ , where  $\Psi(x)$  is as defined above.<sup>20</sup> Letting  $R'(x, r_1, \dots, r_m) = R(x)$ , observe that for every  $(x, r_1, \dots, r_m) \in P$  it holds that  $w \in R'(x, r_1, \dots, r_m)$  if and only if  $\forall i \in [m] D'_{r_i}(x, w) = 1$ . Hence, the promise problem  $(P \cap S', P \setminus S')$  is in the class of promise problems associated with  $\mathcal{UP}$ , and we can apply the PCP of Theorem 3.1 to it. Furthermore, recall that the function  $\Pi'$  (which generates the canonical proof) used to construct the strong canonical PCP system in Theorem 3.1 denoted  $V'$ , assigns to the input  $(x, r_1, \dots, r_m) \in P \cap S'$  the unique proof  $1^t w$  such that  $R(x) = \{w\}$ , where  $t$  is polynomial in  $|(x, r_1, \dots, r_m)|$ . Combining  $\Psi$  with  $V'$  yields a PCP system for  $S$  that, on input  $x$  and oracle access to  $\pi$ , invokes  $V'$  on input  $\Psi(x)$  and provides  $V'$  with oracle access to  $\pi$ . The corresponding verifier, denoted  $V$ , has the following features:

- It (i.e.,  $V$ ) has polynomial proof complexity.  
This feature is inherited from the proof complexity of  $V'$  and the fact that  $|\Psi(x)| = \text{poly}(|x|)$ .
- It satisfies canonical completeness with respect to the function  $\Pi$  such that  $\Pi(x) = 1^t w$  if and only if  $R(x) = \{w\}$ . Indeed,  $\Pi(x) = \Pi'(\Psi(x))$  holds whenever  $\Psi(x) \in P \cap S'$ .  
This is the case because  $\Pr[\Psi(x) \in S'] = 1$  for any  $x \in S$ , and  $1^t w = \Pi'(x, r_1, \dots, r_m)$  and  $R(x) = \{w\}$  hold for any  $(x, r_1, \dots, r_m) \in P \cap S'$ . (We also use the canonical completeness of  $V'$ .)

<sup>20</sup>That is,  $\Psi(x)$  is uniformly distributed over  $\{(x, r_1, \dots, r_{m(|x|)}) : \forall i \in [m(|x|)] r_i \in \{0, 1\}^{p'(|x|+p(|x|))}\}$  and  $m(n) = p(n) + 2$ .

## 30:12 Every Set in $\mathcal{P}$ Is Strongly Testable Under a Suitable Encoding

- It satisfies canonical soundness with respect to the foregoing function  $\Pi$ . Furthermore, invalid proofs are rejected with probability that is proportional to their distance from the valid proof.

This is the case because  $\Pr[\Psi(x) \in P] \geq 3/4$  for any  $x$ , and in that case the strong canonical soundness of  $V'$  beats in. The furthermore clause follows by the fact that  $V'$  is a  $\varrho'$ -strong canonical PCP for  $\varrho'(\alpha) = \alpha/4$ .

Hence,  $V$  is a  $(3\varrho'/4)$ -strong canonical PCP (of polynomial proof complexity) for  $S$ . Note that (typically)  $V$  uses super-logarithmic randomness complexity.<sup>21</sup> ◀

### Strong canonical PCPs of Proximity

Next, we adapt the proof of the positive direction of Theorem 3.1 to the PCPP model.

► **Theorem 3.5** (*UP and strong canonical PCPPs*). *Every set in UP has a strong canonical PCP of Proximity of logarithmic randomness and linear detection probability function. Furthermore, there exists a polynomial-time transformation of NP-witnesses for membership in the set to valid proofs for the resulting PCP.*

Indeed, the positive direction of Theorem 3.1 follows from Theorem 3.5 by applying the latter to the set  $S' = \{C(x) : x \in S\}$ , where  $C$  is a good error correcting code. Note that the claimed PCP system (for  $S$ ) emulates the input-oracle of the PCPP system (for  $S'$ ) by applying  $C$  to its own input  $x$ , and emulating the proof-oracle of the PCPP system by using its own proof-oracle. The canonical soundness of the PCP system (for  $S$ ) follows from the canonical soundness of the PCPP system (for  $S'$ ), since in the case that  $x \notin S$  it holds that the relative distance of  $C(x)$  from the set  $S'$  is a constant.

**Proof.** The construction and its analysis are analogous to those in the proof of Theorem 3.1, except that here we start with a trivial weak-PCPP for the set  $S \in \mathcal{UP}$ , use (parsimonious) gap amplification for PCPPs (see [9, Appendix A.5]), and apply a PCPP version of Proposition 3.2. Details follow.

Starting with the PCPP analogue of Proposition 3.3, we use a similar construction except that we apply it to a fixed 3CNF (which is generated based on the input length only). Recall that the construction consists of two steps: First, we construct a trivial weak-PCPP with unique proofs, and then we apply the gap amplification procedure to it (obtaining a PCPP with unique proofs).

Specifically, in the first step, we reduce the verification of the claim  $x \in S$  to the satisfiability of a fixed 3CNF by an assignment that extends  $x$ , where the formula is derived by the standard Cook–Levin reduction of  $S$  to 3SAT. The fixed formula has main variables  $X$  (which are set by the assignment  $x$ ) and auxiliary variables  $Y$  (which represent the NP-witness for  $x$  as well as intermediate gate-values in the corresponding computation), and the question is whether this formula is satisfiable by an assignment in which  $X = x$ . (Note that when  $x \in S$  there is a unique assignment  $y$  to  $Y$  such that the assignment  $(X, Y) = (x, y)$  satisfies the fixed formula.) The first step is completed by observing that the foregoing formula yields a trivial weak-PCPP (with small but noticeable soundness) that is given oracle access to the input  $x$  (i.e.,  $x$  is the input-oracle) as well as to a proof that corresponds to an assignment to the auxiliary variables. This PCPP has unique valid proofs.

<sup>21</sup> This is inherited from the super-logarithmic length of the proofs employed by the MA system (or, alternatively, from its super-logarithmic randomness complexity). Note that MA systems with logarithmic proof length (resp., logarithmic randomness) exist only for  $\text{co}\mathcal{RP}$  (resp.,  $\mathcal{NP}$ ).

In the second step, we apply the gap amplification procedure, which treats the foregoing PCPP execution as a 2CSP instance such that some variables of the 2CSP are identified with the bits of the input-oracle and the other variables represent various auxiliary values. The set of variables that represents bits of the input-oracle will remain intact throughout the entire process of gap amplification (see [9, Appendix A.5]). Hence, when viewing the resulting 2CSP as a PCPP, the input-oracle of the resulting PCPP equals the input-oracle of the original (trivial) PCPP. Observing (as in the proof of Proposition 3.3) that the gap amplification process maintains the number of valid proofs for each input (see [9, Appendix A.5]), we obtain a PCPP with unique proofs for  $S$ .

Next, we turn to establish a PCPP analogue of Proposition 3.2. This version asserts a transformation of PCPPs with unique proofs to strong canonical PCPPs, and its proof is obtained by a straightforward adaptation of the original (PCP) version.<sup>22</sup> Using the notation of Proposition 3.2, the crux of the analysis is that the pair of oracles  $(x, \pi')$ , where  $x \in \{0, 1\}^n$  (such that  $S \cap \{0, 1\}^n \neq \emptyset$ )<sup>23</sup> and  $\pi' \equiv (w, \pi) \in \{0, 1\}^{(2^{r(n)}-1) \cdot \ell(n) + \ell(n)}$ , is rejected with probability at least

$$\min_{x' \in S \cap \{0, 1\}^n} \{ \max(\Omega(\delta(x, x')), 0.5 \cdot \delta(w, 1^{(2^{r(n)}-1) \cdot \ell(n)}) + 0.5 \cdot 2^{-r(|x|)} \cdot \delta(\pi, \Pi(x'))) \}, \quad (3)$$

where the foregoing lower bound of  $\Omega(\delta(x, x'))$  follows from the soundness of the original PCPP system outlined above. As shown in the proof of Proposition 3.2, it holds that  $\delta(w, 1^{(2^{r(n)}-1) \cdot \ell(n)}) + 2^{-r(|x|)} \cdot \delta(\pi, \Pi(x')) \geq \delta(\pi', \Pi'(x'))$ , which implies that (3) is lower-bounded by  $\Omega(\delta_{\Pi}(x, \pi'))$ .  $\blacktriangleleft$

## 4 The testing result

With strong canonical PCPs of Proximity (as provided by Theorem 3.5) at our disposal, it is quite straightforward to obtain a proximity oblivious tester for a suitable encoding of any set in  $\mathcal{P}$ . Such an encoding incorporates copies of the target object as well as a corresponding PCPP-oracle that attests its membership in the set. To be meaningful, this encoding should be polynomial-time computable and invertible.<sup>24</sup> One may also require that the encoding is “length regular” (i.e., equal length strings have an equal encoding length) and has a constant relative distance, but this seems less essential.

► **Theorem 1.2 (restated).** *For any  $S \in \mathcal{P}$  there exist polynomial-time encoding and decoding algorithms  $E$  and  $D = E^{-1}$  such that the set  $S' \stackrel{\text{def}}{=} \{E(x) : x \in S\}$  has a proximity oblivious tester of linear detection probability. Furthermore,  $|E(x)| = |E(1^{|x|})|$  for every  $x$ , the encoding  $E$  has constant relative distance, and the POT runs in polylogarithmic time and has logarithmic randomness complexity.*

<sup>22</sup> Alternatively, the current version can be derived as a special case of Proposition 5.3.

<sup>23</sup> If  $S \cap \{0, 1\}^n = \emptyset$ , then  $(x, \pi')$  is rejected with probability at least  $\Omega(1)$ , and the claim follows (since in this case  $\delta_{\Pi}(x, \pi') = 1$ ).

<sup>24</sup> The following examples illustrate that restricting the complexity of the encoding is essential for the meaningfulness of Theorem 1.2. Suppose, for example, that for some  $m : \mathbb{N} \rightarrow \mathbb{N}$  it holds that  $|S \cap \{0, 1\}^n| = 2^{m(n)}$ , and consider the length preserving bijection  $E$  that maps the elements of  $S \cap \{0, 1\}^n$  to  $0^{n-m(n)}\{0, 1\}^{m(n)}$ . Then, testing  $\{E(x) : x \in S\}$  amounts to selecting uniformly  $i \in [n - m(n)]$  and checking that the  $i^{\text{th}}$  bit of the  $n$ -bit long input equals 0. More generally, assuming that both  $S$  and  $\bar{S} = \{0, 1\}^* \setminus S$  are infinite, and letting  $\text{idx}_S(w)$  denote the index of  $w \in S$  (e.g., according to the standard lexicographical order), consider the bijection  $E$  such that  $E(x) = y$  if  $x \in S$  (resp.,  $x \in \bar{S}$ ) and  $\text{idx}_S(x) = \text{idx}_T(y)$  (resp.,  $\text{idx}_{\bar{S}}(x) = \text{idx}_{\bar{T}}(y)$ ), where  $T = \cup_{m \in \mathbb{N}} \{0, 1\}^{2m+1}$ . Then, testing  $\{E(x) : x \in S\} = T$  is trivial.

## 30:14 Every Set in $\mathcal{P}$ Is Strongly Testable Under a Suitable Encoding

Recall that proximity oblivious testers (POTs) were defined as having constant query complexity, and that their detection probability function represents a lower bound on their rejection probability as a function of the distance of the tested object from the property.

**Proof.** Let  $C: \{0, 1\}^* \rightarrow \{0, 1\}^*$  be an arbitrary systematic code (i.e.,  $x$  is a prefix of  $C(x)$ ) of relative distance, say,  $1/8$  such that the mapping  $x \mapsto C(x)$  can be computed in polynomial time. Consider a strong canonical PCPP for the set  $C(S) \stackrel{\text{def}}{=} \{C(x) : x \in S\}$ , as guaranteed by Theorem 3.5, and let  $\Pi(C(x))$  denote the (unique) valid proof for  $C(x) \in C(S)$ . The basic idea is to combine the input and proof oracles of the PCPP into a single codeword that will be accessed by the POT as an oracle; in order to maintain the soundness guarantee, it is important that each part of the combined codeword will have approximately the same length. Since the proof-oracle is typically longer, this requires repeating the input-oracle sufficiently many times.

Recalling that  $|\Pi(C(x))| = \ell(|C(x)|)$  for some polynomial  $\ell$ , we proceed to present the claimed algorithms.

**The encoding function  $E$ :** On input  $x \in \{0, 1\}^*$ , we let  $n = |C(x)|$  and  $t = \ell(n)/n$ . Then,  $E(x) = C(x)^t \pi$  such that  $\pi = \Pi(C(x))$  if  $x \in S$ , and  $\pi = 1^{\ell(n)}$  otherwise.

The encoding can be computed in polynomial time, since the canonical proof  $\Pi(C(x))$  can be computed in polynomial time because  $S \in \mathcal{P}$  (and  $C$  is polynomial-time computable). (Formally, the reader may think of  $S$  as being in  $\mathcal{UP}$  by virtue of the witness relation  $R = \{(x, x) : x \in S\}$ , and recall that given an NP-witness one can efficiently obtain the corresponding proof-oracle.)

The code  $C$  and the repetitions are used to create and maintain distance; that is,  $\delta(E(x), E(x')) \geq 0.5 \cdot \delta(C(x), C(x')) \geq 1/16$  for every two distinct  $x, x'$  of equal length.

**The decoding function  $D$ :** On input  $y \in \{0, 1\}^*$ , the algorithm outputs  $x$  if  $|y| = 2\ell(n)$  and  $y = C(x)^{\ell(n)/n} \Pi(C(x))$ , and outputs a special failure symbol otherwise. Specifically, the algorithm first determines  $n = \ell^{-1}(|y|/2)$ , then determines  $k$  such that  $n = |C(1^k)|$ , and finally sets  $x$  as the  $k$ -bit long prefix of  $y$  (and verifies that  $y = C(x)^{\ell(n)/n} \Pi(C(x))$  holds). Note that checking that the suffix of  $y$  is the canonical proof  $\Pi(C(x))$  can be done in polynomial time, since  $x$  is a prefix of  $y$  and  $\Pi \circ C$  is polynomial-time computable.

**The tester  $T$ :** On input  $y \in \{0, 1\}^{2\ell(n)}$ , the tester parses  $y$  into  $(w_1, \dots, w_t, \pi)$  such that  $|w_1| = \dots = |w_t| = n$  and  $|\pi| = \ell(n)$ . It first checks at random that the  $w_i$ 's are all identical to  $w_1$ , by selecting a random  $i \in [t]$  and comparing a random position in  $w_i$  and  $w_1$ . Next, it invokes the (strong canonical) PCPP verifier, providing it with access to the input-oracle  $w_1$  and the proof-oracle  $\pi$ .

We first note that  $D(E(x)) = x$  for every  $x$ . Next, we show that  $T$  is a POT for the set  $S' = \{E(x) : x \in S\}$ . Observe, on the one hand, that for every  $y \in S'$ , it holds that  $y = E(x) = C(x)^t \Pi(C(x))$  for some  $x \in S$ , and the tester  $T$  accepts  $y$  with probability 1 (by virtue of the perfect completeness of the PCPP verifier). On the other hand, turning to the case of  $y \notin S'$  and letting  $y \equiv (w_1, \dots, w_t, \pi) \in \{0, 1\}^{t \cdot n + \ell(n)}$ , we consider three cases (where below, by “far” we mean being at relative distance  $\Omega(\delta_{S'}(y))$ ).

1. If  $(w_1, \dots, w_t)$  is far from  $w_1^t$ , then  $y$  is rejected with proportional probability by the first check of  $T$ .
2. If  $(y$  is close to  $w_1^t$  but)  $w_1$  is far from  $C(S)$ , then  $y$  is rejected with proportional probability by the (strong canonical) PCPP verifier (which is invoked with input-oracle  $w_1$ ).
3. If  $w_1$  is close to  $C(x) \in C(S)$  but  $\pi$  is far from the canonical proof  $\Pi(C(x))$ , then  $y$  is rejected with proportional probability by the (strong canonical) PCPP verifier (which is invoked with input-oracle  $w_1$  and the proof-oracle  $\pi$ ).



(Here we use the fact that if  $w_1$  is close to  $C(x)$ , then it is far from  $C(x')$  for any  $x' \neq x$ . Hence,  $\min_{w'} \{\max(\delta(w_1, w'), \delta(\pi, \Pi(w')))\}$  equals  $\min_{x' \in S} \{\max(\delta(w_1, C(x')), \delta(\pi, \Pi(C(x'))))\}$ , which equals  $\max(1/8, \delta(\pi, \Pi(C(x))))$ .)

Hence,  $T$  is a POT (of linear detection probability) for  $S'$ .  $\blacktriangleleft$

## 5 Directions for further research

The begging question is whether a result like Theorem 1.2 can be proved when using an encoding function of smaller stretch, where the stretch of  $E: \{0, 1\}^* \rightarrow \{0, 1\}^*$  is the function that maps  $n$  to  $|E(1^n)|$ . Specifically, which sets  $S$  satisfy the conclusion of Theorem 1.2 with respect to an encoding of almost linear stretch?

Recalling that our proof of Theorem 1.2 is pivoted at the existence of strong canonical PCPs of Proximity, it is natural to ask which sets have a strong canonical PCP of Proximity of almost-linear proof complexity. We believe that USAT has such a PCP of Proximity, and suggest establishing this conjecture as an open problem.

► **Problem 5.1** (almost-linear length strong canonical PCPPs). *Show that USAT has a strong canonical PCP of Proximity of almost-linear proof complexity. Furthermore, show that valid proofs for this PCPP can be constructed in polynomial-time when given the input formula and a satisfying assignment to it.*

Recall that 3SAT has a PCP of Proximity of almost-linear proof complexity: We refer to the PCPP system of Dinur [8], which builds upon the work of Ben-Sasson and Sudan [6]. A possible route towards resolving Problem 5.1 is to show that this PCPP is a strong canonical PCPP, or can be transformed into such a PCPP at moderate cost (i.e., increasing the proof complexity by a poly-logarithmic factor). We actually believe that such a transformation is needed, since we believe that the PCPP of [8] is almost there (i.e., it satisfies a relaxed form of being strongly canonical (detailed below)), and that the extra step can be made by a generalization (of a PCPP version) of Proposition 3.2.

In order to detail this idea, we need a refinement of Definition 2.2. In this refinement, the rejection probability is not lower-bounded by a function of the maximum of the distances  $\delta(x, x')$  and  $\delta(\pi, \Pi(x))$ , but is rather the maximum of two (potentially) different functions applied to the two distances. Maybe more importantly, we allow these functions to depend also on the input length.

► **Definition 5.2** (Definition 2.2, refined). Let  $\varrho_I, \varrho_P: \mathbb{N} \times [0, 1] \rightarrow [0, 1]$  be monotonically non-decreasing in their second argument such that  $\varrho_I(n, \alpha) = 0$  (resp.,  $\varrho_P(n, \alpha) = 0$ ) if and only if  $\alpha = 0$ . For a set  $S \subseteq \{0, 1\}^*$  and an oracle machine  $V$  that accesses two oracles, we say that  $V$  is a  $(\varrho_I, \varrho_P)$ -strong canonical PCP of Proximity for  $S$  if  $V$  makes a constant number of queries to each of its oracles and there exist functions  $\ell: \mathbb{N} \rightarrow \mathbb{N}$  and  $\Pi: \{0, 1\}^* \rightarrow \{0, 1\}^*$  such that the following conditions hold.

- **Canonical Completeness:** As in Definition 2.2.<sup>25</sup>
- **Strong Canonical Soundness:** For every  $x \in \{0, 1\}^*$  and  $\pi \in \{0, 1\}^{\ell(|x|)}$ , the verifier rejects the pair of oracles  $(x, \pi)$  with probability at least

$$\min_{x' \in \{0, 1\}^{|x|}} \{\max(\varrho_I(|x|, \delta(x, x')), \varrho_P(|x|, \delta(\pi, \Pi(x'))))\}. \quad (4)$$

<sup>25</sup>That is, for every  $x \in S$ , it holds that  $\Pi(x) \in \{0, 1\}^{\ell(|x|)}$  and the verifier always accepts the pair of oracles  $(x, \Pi(x))$ ; i.e.,  $\Pr[V^{x, \Pi(x)}(1^{|x|}) = 1] = 1$ .



### 30:16 Every Set in $\mathcal{P}$ Is Strongly Testable Under a Suitable Encoding

We say that  $V$  is a strong canonical PCP of Proximity for  $S$  if both  $\varrho_{\mathcal{I}}$  and  $\varrho_{\mathcal{P}}$  are oblivious of the length parameter (i.e., if  $\varrho_{\mathcal{I}}(n, \alpha) = \varrho_{\mathcal{I}}'(\alpha)$  for some  $\varrho_{\mathcal{I}}' : [0, 1] \rightarrow [0, 1]$ , and ditto for  $\varrho_{\mathcal{P}}$ ), and say that  $V$  is a semi-strong canonical PCP of Proximity for  $S$  if  $\varrho_{\mathcal{I}}$  is oblivious of the length parameter. As in Definition 2.2,  $\ell$  is called the proof complexity of  $V$ .

Indeed, Definition 2.2 is obtained as a special case by letting  $\varrho_{\mathcal{I}}(n, \alpha) = \varrho_{\mathcal{P}}(n, \alpha) = \varrho(\alpha)$  for  $\varrho : [0, 1] \rightarrow [0, 1]$ . We conjecture that the PCPP system of Dinur [8], which builds on the work of Ben-Sasson and Sudan [6], yields a semi-strong canonical PCP of Proximity of logarithmic randomness and almost linear proof complexity for **USAT**, and that the corresponding function  $\varrho_{\mathcal{P}}$  has the form  $\varrho_{\mathcal{P}}(n, \alpha) = \alpha / \text{poly}(\log n)$ . If this is indeed the case, then using the following transformation, which generalizes Proposition 3.2, yields a strong canonical PCP of Proximity of almost-linear proof complexity for **USAT**, which in turn resolves Problem 5.1.

► **Proposition 5.3** (deriving strong canonical PPCPs from semi-strong ones). *Let  $V$  be a  $(\varrho_{\mathcal{I}}, \varrho_{\mathcal{P}})$ -strong canonical PCPP system of logarithmic randomness complexity and proof complexity  $\ell$  for  $S$ , and suppose that  $\varrho_{\mathcal{I}}(n, \alpha) = \varrho(\alpha) \leq \alpha$  and  $\varrho_{\mathcal{P}}(n, \alpha) = \varrho'_{\mathcal{P}}(n) \cdot \alpha$  for some  $\varrho'_{\mathcal{P}} : \mathbb{N} \rightarrow (0, 1]$ . Then, there exists a  $\varrho$ -strong canonical PCP of logarithmic randomness and proof complexity  $\ell / \varrho'_{\mathcal{P}}$  for  $S$ . Furthermore, there exists a polynomial-time transformation of valid proofs with respect to  $V$  to valid proofs for the resulting PCP.*

Note that the PCPP analogue of Proposition 3.2 follows as a special case by observing that any canonical PCPP system (i.e., one that has unique valid proofs) of randomness complexity  $r$  is an  $(\varrho_{\mathcal{I}}, \varrho_{\mathcal{P}})$ -strong canonical PCPP, where  $\varrho_{\mathcal{I}}$  is oblivious of the length parameter and  $\varrho_{\mathcal{P}}(n, \alpha) = 2^{-r(n)}$ .

**Proof.** We observe that the proof of Proposition 3.2 can be adapted and generalized as follows. Again, we may assume, without loss of generality, that  $V$  has polynomial proof complexity, and let  $t(n) = 1 / \varrho'_{\mathcal{P}}(n)$ . Letting  $\Pi$  be as in Definition 2.2, we define  $\Pi'(x) = 1^{(t(n)-1) \cdot \ell(x)} \Pi(x)$  if  $\Pi(x) \neq \lambda$ , and  $\Pi'(x) = \lambda$  otherwise. Analogously to the proof of Proposition 3.2, we consider the following verifier, which given oracle access to an input  $x \in \{0, 1\}^n$  and an alleged proof  $\pi' \in \{0, 1\}^{t(n) \cdot \ell(n)}$ , selects uniformly at random one of the following two tests and performs it.

1. The verifier selects at random  $i \in [(t(n) - 1) \cdot \ell(n)]$ , and accepts if and only if the  $i^{\text{th}}$  bit of  $\pi'$  equals 1.
2. The verifier invokes  $V$  on input  $x$  and the  $\ell(n)$ -bit long suffix of  $\pi'$ . That is,  $V$ 's queries to the input-oracle are answered by the input-oracle of the new verifier, whereas  $V$ 's queries to the proof-oracle are answered by accessing the suffix of the proof-oracle of the new verifier (i.e., query  $i \in [\ell(n)]$  is answered by querying the location  $(t(n) - 1) \cdot \ell(n) + i$  in  $\pi'$ ).

Turning to the analysis, we first note that if  $S \cap \{0, 1\}^n = \emptyset$ , then  $(x, \pi')$  is rejected with probability at least  $\delta(1) = \Omega(1)$ , and the claim follows. Hence, we may assume that  $S \cap \{0, 1\}^n \neq \emptyset$ . Letting  $\pi' \equiv (w, \pi) \in \{0, 1\}^{(t(n)-1) \cdot \ell(n) + \ell(n)}$ , we infer that the pair of oracles  $(x, \pi') \in \{0, 1\}^n \times \{0, 1\}^{t(n) \cdot \ell(n)}$  is rejected with probability at least

$$\min_{x' \in S \cap \{0, 1\}^n} \{ \max(\varrho(\delta(x, x')), 0.5 \cdot \delta(w, 1^{(t(n)-1) \cdot \ell(n)}) + 0.5 \cdot \varrho'_{\mathcal{P}}(n) \cdot \delta(\pi, \Pi(x'))) \}.$$

Observing that

$$\begin{aligned} & \delta(w, 1^{(t(n)-1) \cdot \ell(n)}) + \varrho'_{\mathcal{P}}(n) \cdot \delta(\pi, \Pi(x')) \\ & \geq \delta(w\pi, 1^{t(n)-1} \cdot \ell(n) \pi) + \varrho'_{\mathcal{P}}(n) \cdot t(n) \cdot \delta(1^{t(n)-1} \cdot \ell(n) \pi, 1^{t(n)-1} \cdot \ell(n) \Pi(x')) \\ & = \delta(\pi', 1^{t(n)-1} \cdot \ell(n) \pi) + \delta(1^{t(n)-1} \cdot \ell(n) \pi, \Pi'(x')) \\ & \geq \delta(\pi', \Pi'(x')), \end{aligned}$$

it follows that  $(x, \pi')$  is rejected with probability at least

$$\min_{x' \in S \cap \{0,1\}^n} \{\max(\varrho(\delta(x, x')), 0.5 \cdot \delta(\pi', \Pi'(x')))\}.$$

Using  $\varrho(\alpha) \leq \alpha$ , it follows that the new verifier is a  $0.5\varrho$ -strong canonical PCPP of proof complexity  $\ell/\varrho'_p$ . ◀

---

## References

- 1 N. Alon, E. Fischer, M. Krivelevich, and M. Szegedy. Efficient testing of large graphs. *Combinatorica*, 20:451–476, 2000.
- 2 N. Alon, E. Fischer, I. Newman, and A. Shapira. A combinatorial characterization of the testable graph properties: It’s all about regularity. *STOC*, pages 251–260, 2006.
- 3 S. Arora, C. Lund, R. Motwani, M. Sudan, and M. Szegedy. Proof verification and intractability of approximation problems. *JACM*, 45:501–555, 1998.
- 4 S. Arora and S. Safra. Probabilistic checkable proofs: A new characterization of NP. *JACM*, 45:70–122, 1998.
- 5 E. Ben-Sasson, O. Goldreich, P. Harsha, M. Sudan, and S. Vadhan. Robust pcps of proximity, shorter pcps, and applications to coding. *SICOMP*, 36(4):889–974, 2006.
- 6 E. Ben-Sasson and M. Sudan. Short pcps with polylog query complexity. *SICOMP*, 38(2):551–607, 2008.
- 7 M. Blum, M. Luby, and R. Rubinfeld. Self-testing/correcting with applications to numerical problems. *JCSS*, 47(3):549–595, 1993.
- 8 I. Dinur. The pcp theorem by gap amplification. *JACM*, 54:3, 2007.
- 9 I. Dinur, O. Goldreich, and T. Gur. Every set in p is strongly testable under a suitable encoding. Technical report, in ECCC TR18-050, 2018.
- 10 I. Dinur and O. Reingold. Assignment-testers: Towards a combinatorial proof of the pcp-theorem. *FOCS*, 45, 2004.
- 11 O. Goldreich. *Computational Complexity: A Conceptual Perspective*. Cambridge University Press, 2008.
- 12 O. Goldreich. *Introduction to Property Testing*. Cambridge University Press, 2017.
- 13 O. Goldreich, S. Goldwasser, and D. Ron. Property testing and its connection to learning and approximation. *JACM*, pages 653–750, 1998.
- 14 O. Goldreich, T. Gur, and I. Komargodski. Strong locally testable codes with relaxed local decoders. *CCC*, 30:1–41, 2015.
- 15 O. Goldreich, M. Krivelevich, I. Newman, and E. Rozenberg. Hierarchy theorems for property testing. *Computational Complexity*, 21(1):129–192, 2012.
- 16 O. Goldreich and D. Ron. Property testing in bounded degree graphs. *Algorithmica*, pages 302–343, 2002.
- 17 O. Goldreich and D. Ron. On proximity oblivious testing. *SICOMP*, 40(2):534–566, 2011.
- 18 O. Goldreich and I. Shinkar. Two-sided error proximity oblivious testing. *RSA*, 48(2):341–383, 2016.
- 19 O. Goldreich and M. Sudan. Locally testable codes and pcps of almost-linear length. *JACM*, 53(4):558–655, 2006.
- 20 T. Gur, G. Ramnarayan, and R. Rothblum. *Relaxed Locally Correctable Codes*. ITCS. ECCC, 2018.
- 21 T. Gur and R. Rothblum. Non-interactive proofs of proximity. *Computational Complexity*, 2018.



# Alea lacta Est: Auctions, Persuasion, Interim Rules, and Dice

**Shaddin Dughmi**

University of Southern California, Los Angeles, CA, USA  
shaddin@usc.edu

**David Kempe**

University of Southern California, Los Angeles, CA, USA  
David.M.Kempe@gmail.com

**Ruixin Qiang**

University of Southern California, Los Angeles, CA, USA  
rqiang@usc.edu

---

## Abstract

To select a subset of samples or “winners” from a population of candidates, order sampling [20] and the  $k$ -unit Myerson auction [17] share a common scheme: assign a (random) score to each candidate, then select the  $k$  candidates with the highest scores. We study a generalization of both order sampling and Myerson’s allocation rule, called *winner-selecting dice*. The setting for winner-selecting dice is similar to auctions with feasibility constraints: candidates have random types drawn from independent prior distributions, and the winner set must be feasible subject to certain constraints. Dice (distributions over scores) are assigned to each type, and winners are selected to maximize the sum of the dice rolls, subject to the feasibility constraints. We examine the existence of winner-selecting dice that implement prescribed probabilities of winning (i.e., an interim rule) for all types.

Our first result shows that when the feasibility constraint is a matroid, then for any feasible interim rule, there always exist winner-selecting dice that implement it. Unfortunately, our proof does not yield an efficient algorithm for constructing the dice. In the special case of a 1-uniform matroid, i.e., only one winner can be selected, we give an efficient algorithm that constructs winner-selecting dice for any feasible interim rule. Furthermore, when the types of the candidates are drawn in an i.i.d. manner and the interim rule is symmetric across candidates, unsurprisingly, an algorithm can efficiently construct symmetric dice that only depend on the type but not the identity of the candidate.

One may ask whether we can extend our result to “second-order” interim rules, which not only specify the winning probability of a type, but also the winning probability conditioning on each other candidate’s type. We show that our result does not extend, by exhibiting an instance of *Bayesian persuasion* whose optimal scheme is equivalent to a second-order interim rule, but which does not admit any dice-based implementation.

**2012 ACM Subject Classification** Theory of computation → Algorithmic game theory

**Keywords and phrases** Interim rule, order sampling, virtual value function, Border’s theorem

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2019.31

**Related Version** A full version of the paper is available at <https://arxiv.org/abs/1811.11417>.

**Funding** Work supported in part by NSF Grant CCF-1423618.



© Shaddin Dughmi, David Kempe, and Ruixin Qiang;  
licensed under Creative Commons License CC-BY  
10th Innovations in Theoretical Computer Science (ITCS 2019).  
Editor: Avrim Blum; Article No. 31; pp. 31:1–31:20



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

## 1 Introduction

Economic design often features scenarios in which choices must be made based on stochastic inputs. In auction design, bidders drawn from a population interact with an auction mechanism, and the mechanism must then choose winners and losers of the auction. In information structure design, a principal observes information pertinent to the various actions available to one or more decision makers, and must use this information to recommend actions to the decision makers. In such settings, an important framing device is the notion of an *interim rule* (also called a *reduced form*) of an (ex-post) winner selection rule, summarizing the probability for each candidate to be selected.

We focus on the simplest and most natural class of such decision making scenarios, one which includes auctions and Bayesian persuasion [12] as special cases. In a *winner selection* environment, there is a set of *candidates*  $\mathcal{C}$ , each equipped with a random attribute known as its *type*. A *winner selection rule* is a randomized function (or algorithm) which maps each profile of types, one per candidate, to a choice of winning candidates, subject to the requirement that the set of winners must belong to a specified family  $\mathcal{I} \subseteq 2^{\mathcal{C}}$  of feasible sets. The winner selection rule is also referred to as an ex-post rule since it specifies the winning probabilities conditioned on every realized type profile. In auctions, candidates correspond to bidders, and a winner selection rule is an allocation rule of the auction. In Bayesian persuasion, candidates correspond to actions available to a decision maker, and a winner selection rule corresponds to a persuasion scheme used by a principal to recommend one of the actions to the decision maker. We restrict our attention to winner selection scenarios in which the types of different candidates are independently distributed.

We distinguish two classes of interim rules: *first-order* and *second-order*. The former is the traditional notion from auction theory, while the latter is the notion better suited for persuasion. A *first-order interim rule* specifies, for each candidate  $i$  and type  $t$  of candidate  $i$ , the conditional probability of  $i$  winning given that his type is  $t$ . A *second-order interim rule* specifies more information: for each pair of candidates  $i, j$  and type  $t$  of candidate  $j$ , it specifies the conditional probability of  $i$  winning given that  $j$  has type  $t$ . First-order interim rules, when combined with a payment rule, suffice for evaluating the welfare, revenue, and incentive-compatibility of a single-item auction. For Bayesian persuasion, second-order interim rules are needed for evaluating the incentive constraints of a persuasion scheme.

Our motivation for studying winner selection at this level of generality stems from the success of Myerson's [17] famous and elegant characterization of revenue-optimal single-item auctions when bidder type distributions are independent. In that special case, Myerson showed that the optimal *single-item* auction features a particularly structured winner-selection rule: each type is associated with a virtual value, and given a profile of reported types, the rule selects the bidder with the highest (non-negative) virtual value as the winner.

Interestingly, order sampling [20] works in a similar way as Myerson's auction, in the special case when the feasible sets are sets of  $k$  or fewer candidates, and each candidate has only one type. Order sampling assigns each candidate a random score variable (a die). The winners are the candidates with the  $k$  highest score variables. In the language of order sampling, virtual value functions define a single-sided die for each type. Unlike Myerson's auction, there is no notion of "revenue" or "incentive constraints" in order sampling. The task is simply to find dice that will induce a prescribed first-order interim rule.

As a generalization of order sampling and Myerson's virtual-value approach, a *dice-based winner-selection rule* assigns each type a die, and selects the feasible set of winners maximizing the sum of the dice rolls. *We explore the extent to which dice-based rules are applicable beyond*

*single-parameter auctions or single-type environments, to winner selection with independent type distributions under more complex constraints.* In particular, we examine whether dice-based winner-selection rules exist for winner selection subject to matroid constraints and for Bayesian persuasion.

## 1.1 Our Results

As mentioned previously, all of our results are restricted to settings in which the candidates' type distributions are independent. It follows from Myerson's characterization that every first-order interim rule *corresponding to some optimal auction* admits a dice-based implementation.<sup>1</sup> Our main result (in Section 3) is an existential proof showing that *every* feasible first-order interim rule with respect to a matroid constraint admits a dice-based implementation. This illustrates that the structure revealed by Myerson's characterization is more general, and applies to other settings in which only first-order interim information is relevant. For example, single-item auctions with (public or private) budgets are such a setting (see, e.g., [19]). Our result also provides a generalization of order sampling from the  $k$ -winner setting to the general matroid, multi-type setting.

Beyond the existential proof of dice-based implementations, we show (in Section 4) that for single-winner environments, an algorithm can construct the dice-based rule *efficiently*. When the types are identically distributed, we also constructively show (in the full version) that every first-order interim rule which is symmetric across candidates admits a symmetric dice implementation; i.e., different candidates have the same die for the same type. This is consistent with Myerson's symmetric characterization of optimal single-item auctions with i.i.d. bidders, and generalizes it to any other first-order single-winner selection setting in which candidates are identical. Single-item auctions with identically distributed budgeted bidders are such a setting, and a symmetric dice-based implementation of the optimal allocation rule was already known from [19].

For single-winner cases, we also show the converse direction: how to efficiently compute the first-order interim rule of a given dice-based winner selection rule. In effect, these results show that collections of dice are a computationally equivalent description of single-winner first-order interim information. This implies a kind of equivalence between the two dominant approaches for mechanism design: the Myersonian approach based on virtual values (i.e., dice), and the Borderian approach based on optimization over interim rules.

In an attempt to leverage the same kinds of insights for Bayesian persuasion, we examine (in Sections 5 and 6) the dice implementability of second-order interim rules. When the candidate type distributions are non-identical, we show an impossibility result. We construct an instance of Bayesian persuasion with independently distributed non-identical actions, and show that no optimal persuasion scheme for this instance can be implemented by dice. Since second-order interim rules are sufficient for evaluating the objective and constraints of Bayesian persuasion, this implies that there exist second-order interim rules which are not dice-implementable. This rules out the Myersonian approach for characterizing and computing optimal schemes for Bayesian persuasion with independent non-identical actions, complementing the negative result of [8] which rules out the Borderian approach for the same problem.

---

<sup>1</sup> As we show in the full version, there are interim rules which are not optimal for any auction.

Our impossibility result disappears when the actions are i.i.d., since second-order interim rules collapse to first-order interim rules in symmetric settings. In particular, as we show in Section 6, our results for first-order interim rules, combined with those of [8], imply that Bayesian persuasion with i.i.d. actions admits an optimal dice-based scheme, which can be computed efficiently.

## 1.2 Additional Discussion of Related Work

Myerson [17] was the first to characterize revenue-optimal single-item auctions; this characterization extends to single-parameter mechanism design settings more generally (see, e.g., [11]). The (first-order) interim rule of an auction, also known as its reduced form, was first studied by Maskin and Riley [14] and Matthews [15]. The inequalities characterizing the space of feasible interim rules were described by Border [3, 4]. Border’s analytically tractable characterization of feasible interim rules has served as a fruitful framework for mechanism design, since an optimal auction can be viewed as the solution of a linear program over the space of interim rules. Moreover, this characterization has enabled the design of efficient algorithms for recognizing interim rules and optimizing over them, by Cai et al. [5] and Alaei et al. [2]. This line of work has served as a foundation for much of the recent literature on Bayesian algorithmic mechanism design in multi-parameter settings.

It is important to contrast our dice-based rule with the characterization of Cai et al. [5]. In particular, the results of Cai et al. [5] imply that every first-order interim rule can be efficiently implemented as a distribution over virtual value maximizers. In our language, this implies the existence of an efficiently computable dice-based implementation *in which the dice may be arbitrarily correlated*. Our result, in contrast, efficiently computes a family of *independent dice* implementing any given first-order interim rule in single-winner settings, and shows the existence of a dice-based rule in matroid settings. This is consistent with Myerson’s characterization, in which virtual values are drawn independently.

Alaei et al. [2] also studied winner-selection environments, under the different name “service based environments.” For single-winner settings, they proposed a mechanism called stochastic sequential allocation (SSA). The mechanism also implements any feasible first-order interim rule, by creating a token of winning and transferring the token sequentially from one candidate to another, with probabilities defined by an efficiently computed transition table. Dice can be considered as the special case of SSA in which the transition probabilities are independent of the current owner of the token.

As another motivation for our focus on dice-based rules, order sampling studies how to sample  $k$  winners from  $n$  candidates with given inclusion probabilities (i.e., implement an interim rule), by assigning a random score variable (die) to each candidate. Rosén [20] showed that parameterized Pareto distributions can be used to implement a given interim rule asymptotically. Aires et al. [1] proved the existence of an order sampling scheme that exactly implements any feasible interim rule. Our existential proof is a generalization of the proof of [1] to settings with multiple types and matroid constraints.

The Bayesian persuasion model is due to Kamenica and Gentzkow [12], and is the most influential model in the general space of information structure design (see the survey by Dughmi [7] for references). Bayesian persuasion was examined algorithmically by Dughmi and Xu [8], who observed its connection to auction theory and interim rules, and examined the computational complexity of optimal schemes through the lens of optimization over interim rules.



Of particular relevance to our work is the negative result of Dughmi and Xu [8] for Bayesian persuasion with independent non-identical actions: it is  $\#P$ -hard to compute the interim rule (first- or second-order) of the optimal scheme, or more simply even the sender's optimal utility. Most notable about this result is what it *does not* rule out: an algorithm implementing the optimal persuasion scheme “on the fly,” in the sense that it efficiently samples the optimal scheme's (randomized) recommendation when given as input the profile of action types. Stated differently, the negative result of Dughmi and Xu [8] merely rules out the Borderian approach for this problem, leaving other approaches – such as the Myersonian one – viable as a means of obtaining an efficient “on the fly” implementation. This would not be unprecedented: Gopalan et al. [9] exhibit a simple single-parameter auction setting for which the optimal interim rule is  $\#P$  hard to compute, yet Myerson's virtual values can be sampled efficiently and used to efficiently implement the optimal auction. Our negative result in Section 6 rules out such good fortune for Bayesian persuasion with independent non-identical actions: there does not exist a (Myersonian) dice-based implementation of the optimal persuasion scheme in general.

## 2 Preliminaries

### 2.1 Winner Selection

Consider choosing a set of winners from among  $n$  candidates. Each candidate  $i$  has a type  $t_i \in T_i$ , drawn independently from a distribution  $f_i$ . A winner-selection rule  $\mathcal{A}$  maps each type profile  $\mathbf{t} = (t_1, \dots, t_n)$ , possibly randomly, to one of a prescribed family of feasible sets  $\mathcal{I} \subseteq 2^{[n]}$ . When  $i \in \mathcal{A}(\mathbf{t})$ , we refer to  $i$  as a *winning candidate*, and to  $t_i$  as his *winning type*. Writing  $f = f_1 \times \dots \times f_n$  for the (independent) joint type distribution, we also refer to  $(f, \mathcal{I})$  as the *winner-selection environment*. When  $\mathcal{I}$  is the family of singletons, as in the setting of the single item auction, we call  $(f, \mathcal{I})$  a *single-winner environment*.

This general setup captures the allocation rules of general auctions with independent unit-demand buyers, albeit without specifying payment rules or imposing incentive constraints. Moreover, it captures Bayesian persuasion with independent action payoffs, albeit without enforcing persuasiveness (also called *obedience*) constraints.

### 2.2 Matroids

In this paper, we focus on settings in which the feasible sets  $\mathcal{I}$  are the independent sets of a matroid. We use the standard definition of a matroid  $\mathcal{M}$  as a pair  $(E, \mathcal{I})$ , where  $E$  is the *ground set* and  $\mathcal{I} \subseteq 2^E$  is a family of so-called *independent sets*, satisfying the three matroid axioms. We also use the standard definitions of a *circuit* and *rank function*  $r_{\mathcal{M}} : 2^E \rightarrow \mathbb{N}$ . The *restriction*  $\mathcal{M}|_S$  of  $\mathcal{M} = (E, \mathcal{I})$  to some  $S \subseteq E$  is the matroid  $(E, \mathcal{I} \cap 2^S)$ .<sup>2</sup> For details on matroids, we refer the reader to Oxley [18].

A matroid  $\mathcal{M} = (E, \mathcal{I})$  is *separable* if it is a *direct sum* of two matroids  $\mathcal{M}_1 = (E_1, \mathcal{I}_1)$  and  $\mathcal{M}_2 = (E_2, \mathcal{I}_2)$ . Namely,  $E = E_1 \uplus E_2$ ,  $\mathcal{I} = \{A \cup B \mid A \in \mathcal{I}_1, B \in \mathcal{I}_2\}$ . Note that if  $\mathcal{M}$  is non-separable, then  $r_{\mathcal{M}}(E) < |E|$ ; otherwise  $\mathcal{M}$  is the direct sum of singleton matroids. We use the following theorem.

► **Theorem 1** (Whitney [21]). (1) When  $\mathcal{M} = (E, \mathcal{I})$  is a non-separable matroid, for every  $a, b \in E$ , there is a circuit containing both  $a$  and  $b$ . (2) Any separable matroid  $\mathcal{M}$  is a direct sum of two or more non-separable matroids called the components of  $\mathcal{M}$ .

<sup>2</sup> Note that we deviate slightly from the standard definition in that we do not restrict the ground set.

In most of the remainder of the paper, we focus on winner selection environments  $(f, \mathcal{I})$  where  $\mathcal{I}$  is the family of independent sets of a matroid  $\mathcal{M}$ . We therefore also use  $(f, \mathcal{M})$  to denote the environment.

## 2.3 Interim Rules and Border's Theorem

A (first-order) interim rule  $\mathbf{x}$  specifies the winning probability  $x_i(t) \in [0, 1]$  for all  $i \in [n], t \in T_i$  in an environment  $(f, \mathcal{I})$ . More precisely, we say that a winner-selection rule  $\mathcal{A}$  implements the interim rule  $\mathbf{x}$  for a prior  $f$  if it satisfies the following: if the type profile  $\mathbf{t} = (t_1, \dots, t_n)$  is drawn from the prior distribution  $f = f_1 \times \dots \times f_n$ , then  $\Pr[i \in \mathcal{A}(\mathbf{t}) \mid t_i = t] = x_i(t)$ . An interim rule is *feasible* (or *implementable*) within an environment  $(f, \mathcal{I})$  if there is a winner-selection rule implementing it that always outputs an independent set of  $\mathcal{I}$ .

### 2.3.1 Border's theorem and implications for single-winner environments

The following theorem characterizes the space of feasible interim rules for single-winner settings.

► **Theorem 2** (Border [3, 4]). *An interim rule  $\mathbf{x}$  is feasible for a single-winner setting if and only if for all possible type subsets  $S_1 \subseteq T_1, S_2 \subseteq T_2, \dots, S_n \subseteq T_n$ ,*

$$\sum_{i=1}^n \sum_{t \in S_i} f_i(t) x_i(t) \leq 1 - \prod_{i=1}^n \left( 1 - \sum_{t \in S_i} f_i(t) \right). \quad (1)$$

The following result leverages Theorem 2 to show that efficient algorithms exist for checking the feasibility of an interim rule, and for implementing a feasible interim rule.

► **Theorem 3** ([5, 2]). *Given explicitly represented priors  $f_1, \dots, f_n$  and an interim rule  $\mathbf{x}$  in a single-winner setting, the feasibility of  $\mathbf{x}$  can be checked in time polynomial in the number of candidates and types. Moreover, given a feasible interim rule  $\mathbf{x}$ , an algorithm can find a winner-selection rule implementing  $\mathbf{x}$  in time polynomial in the number of candidates and types.*

In our efficient construction for single-winner settings, we utilize a structural result which shows that checking only a subset of Border's constraints suffices [3, 16, 5]. This subset of constraints can be identified efficiently.

► **Theorem 4** (Theorem 4 of [5]). *An interim rule  $\mathbf{x}$  is feasible for a single-winner setting if and only if for all possible  $\alpha \in [0, 1]$ , the sets  $S_i(\alpha) = \{t \in T_i \mid x_i(t) > \alpha\}$  satisfy the following Border's constraint:*

$$\sum_{i=1}^n \sum_{t \in S_i(\alpha)} f_i(t) x_i(t) \leq 1 - \prod_{i=1}^n \left( 1 - \sum_{t \in S_i(\alpha)} f_i(t) \right).$$

When the candidates' type distributions are i.i.d., i.e.,  $T_i$  and  $f_i$  are the same for all candidates  $i$ , it is typically sufficient to restrict attention to *symmetric* interim rules. For such rules,  $x_i(t)$  is equal for all candidates  $i$ . In the i.i.d. setting, we therefore notationally omit the dependence on the candidate and let  $T$  refer to the common type set of all candidates,  $f$  to the candidates' (common) type distribution, and  $x(t)$  to the probability that a particular candidate wins conditioned on having type  $t$ . In the i.i.d. setting, only the symmetric constraints from Theorem 2 suffice to characterize feasibility [3]; namely,

$$n \cdot \sum_{t \in S} f(t) x(t) \leq 1 - \left( 1 - \sum_{t \in S} f(t) \right)^n, \quad (2)$$

for all  $S \subseteq T$ . Theorem 4 then implies that it suffices to check Inequality (2) for sets of the form  $S(\alpha) = \{t \in T : x(t) > \alpha\}$  with  $\alpha \in [0, 1]$ .

### 2.3.2 Border’s Theorem for matroid environments

For general settings with matroid constraints, Alaei et al. [2] established the following generalized “Border’s Theorem.”

► **Theorem 5** (Theorem 7 of [2]). *Let  $\nu$  map each type  $t$  to the (unique) candidate  $i$  with  $t \in T_i$ . An interim rule  $\mathbf{x}$  is feasible within an environment  $(f, \mathcal{M})$  if and only if for all possible type subsets  $S_1 \subseteq T_1, S_2 \subseteq T_2, \dots, S_n \subseteq T_n$ ,  $\sum_{i=1}^n \sum_{t \in S_i} f_i(t)x_i(t) \leq \mathbb{E}_{\mathbf{t} \sim f} [r_{\mathcal{M}}(\nu(\mathbf{t} \cap S))]$ , where  $S = \bigcup_{i=1}^n S_i$ .*

In later sections, we omit the function  $\nu$ , and for any type set  $S$  just write  $r_{\mathcal{M}}(S)$  instead of  $r_{\mathcal{M}}(\nu(S))$ .

## 2.4 Winner-Selecting Dice

We study winner-selection rules based on *dice*, as a generalization of order sampling to multiple types and general constraints. A *dice-based rule* fixes, for each type  $t \in T_i$ , a distribution  $D_{i,t}$  over real numbers, which we call a *die*. Given as input the type profile  $\mathbf{t} = (t_1, \dots, t_n)$ , the rule independently draws a score  $v_i \sim D_{i,t_i}$  for each candidate  $i$  by “rolling his die;” it then selects the feasible set of candidates maximizing the sum of scores as the winner set, breaking ties with a predefined rule. In this paper, we will mainly discuss matroid feasibility constraints, for which a feasible set maximizing the sum of scores can be found by a simple greedy algorithm: candidates are added to the winner set in decreasing order of their scores, breaking ties uniformly at random, as long as the new winner set is still an independent set of the matroid and their scores are positive. When candidates have the same type sets, we call a dice-based rule *symmetric* if  $D_{i,t}$  is the same for all  $i$ .

Myerson’s optimal auction is a dice-based winner-selection rule. In Myerson’s nomenclature,  $\bar{v}_i$  is candidate  $i$ ’s *virtual value*, and  $D_{i,t}$  is a single-sided die with the virtual value.

Let  $T$  be the set of all types of all candidates and  $\mathcal{D} = (D_t)_{t \in T}$  be a vector of dice, one per type. Given an interim rule  $\mathbf{x}$ , and a winner-selection environment  $(f, \mathcal{I})$ , we say that  $\mathcal{D}$  *implements*  $\mathbf{x}$ , or  $\mathcal{D}$  describes *winner-selecting dice* for  $\mathbf{x}$  in  $(f, \mathcal{I})$ , if the dice-based rule given by  $\mathcal{D}$  implements  $\mathbf{x}$  within the environment  $(f, \mathcal{I})$ .

## 2.5 Second-order Interim Rules

A (first-order) interim rule, as defined in Section 2.3, specifies, for each candidate  $i$ , the conditional type distribution of  $i$  in the event that  $i$  is chosen as the winner. We define a *second-order interim rule*<sup>3</sup> which maintains strictly more information, as needed for describing the incentive constraints of Bayesian persuasion. Such a rule specifies, for each pair of candidates  $i$  and  $i'$  (where  $i'$  may or may not be equal to  $i$ ), the conditional type distribution of  $i'$  in the event that  $i$  is chosen as the winner. Formally, a second-order interim

<sup>3</sup> Our notion of second-order interim rules is different from the notion defined in [6]. Because Cai et al. [6] consider correlation in types, their notion of second-order interim rules is aimed at capturing the allocation dependencies arising through such type correlation, rather than solely through the mechanism’s choice.

rule  $\mathbf{X}$  specifies  $x_{i,i',t} \in [0, 1]$  for each pair of candidates  $i, i' \in [n]$ , and type  $t \in T_{i'}$ . We say that a winner-selection rule  $\mathcal{A}$  *implements*  $\mathbf{X}$  for a prior  $f$  if it satisfies the following: if the type profile  $\mathbf{t} = (t_1, \dots, t_n)$  is drawn from the prior distribution  $f = f_1 \times \dots \times f_n$ , then  $\Pr[i \in \mathcal{A}(\mathbf{t}) \mid t_{i'} = t] = x_{i,i',t}$ . A second-order interim rule is *feasible* if there is a winner selection rule implementing it.

### 3 Existence of Dice Implementation for Matroids

In this section, we outline the proof of our first theorem:

► **Theorem 6.** *Let  $(f, \mathcal{M})$  be a matroid winner selection-environment with a total of  $m$  types, and let  $\mathbf{x}$  be an interim rule that is feasible within  $(f, \mathcal{M})$ . There exist winner-selecting dice  $\mathcal{D}$ , each of which has at most  $m + 1$  faces, which implement  $\mathbf{x}$ .*

The proof consists of two parts. First, we generalize the result of [1], which showed the existence of continuous winner-selecting dice for feasible interim rules for a  $k$ -uniform matroid with fixed types, to general matroids and multiple types. Second, we convert the continuous dice to dice with at most  $m + 1$  faces each, while keeping the interim probabilities unchanged.

#### 3.1 Continuous Winner-Selecting Dice

► **Theorem 7.** *Let  $\mathcal{M}$  be a matroid, and  $\mathbf{x}$  a feasible interim rule within the winner-selection environment  $(f, \mathcal{M})$ . There exist winner-selecting dice  $\mathcal{D}$  over  $\mathbb{R}$  that implement  $\mathbf{x}$  in  $(f, \mathcal{M})$ .*

We assume without loss of generality that the candidates' type sets are disjoint, and use  $T = \bigsqcup_{i=1}^n T_i$  to denote the set of all types. We use  $f(t)$  and  $x(t)$  as shorthand for  $f_i(t)$  and  $x_i(t)$ , where  $i = \nu(t)$  is the candidate for whom  $t \in T_i$ . Moreover, given a set of types  $S \subseteq T$ , we write  $S_i = S \cap T_i$ . Recall the Border constraints

$$\sum_{i=1}^n \sum_{t \in S_i} f_i(t)x_i(t) \leq R(S), \quad (3)$$

where  $R(S) = \mathbb{E}_{\mathbf{t} \sim f} [r_{\mathcal{M}}(\mathbf{t} \cap S)]$  is the expected rank of types in  $S$  which show up, a submodular function over the type set  $T$ . The Border constraints can therefore be interpreted as follows: An interim rule  $\mathbf{x} : T \rightarrow [0, 1]$  is feasible for  $f$  and  $\mathcal{M}$  if and only if  $\tilde{\mathbf{x}}$  is in the polymatroid given by  $R(S)$ , where  $\tilde{\mathbf{x}}(t) := f(t)x(t)$ . Equivalently,  $\mathbf{x}$  is feasible if and only if the submodular *slack function*  $\sigma(S) = R(S) - \sum_{t \in S} f(t)x(t)$  is non-negative everywhere.

When  $\mathbf{x}$  is feasible, we call a set  $S \subseteq T$  *tight* for  $(f, \mathbf{x}, \mathcal{M})$  if the Border constraint (3) corresponding to  $S$  is tight at  $\mathbf{x}$ , i.e.,  $\sum_{i=1}^n \sum_{t \in S_i} f_i(t)x_i(t) = R(S)$ . By definition,  $S = \emptyset$  is always tight. The family of tight sets, being the family of minimizers of the submodular slack function, forms a lattice: the intersection and the union of two tight sets is a tight set.

► **Remark.** The tightness of a set  $S$  means that the expected number of winners from  $S$  equals the expected rank of types in  $S$  which show up. In other words,  $S$  is tight if and only if a maximum independent subset of  $\mathbf{t} \cap S$  is always selected as winners.

By the preceding remark, the types in minimal non-empty tight sets need to be treated preferentially, i.e., assigned higher faces on their dice, compared to types outside them. Because they play such an important role, we define them as *barrier sets*. Formally, we define the set of *active* types  $T^+ = \{t \in T : f(t)x(t) > 0\}$  to be the types who win with positive probability. Barrier sets are subsets of  $T^+$ . If there is at least one non-empty tight set of active types, we define the barrier sets as the (inclusion-wise) minimal such sets. Otherwise, we designate the entire set  $T^+$  of active types as the unique barrier set.

To prove Theorem 7, we first assume that the matroid  $\mathcal{M}$  is non-separable. Separable matroids will be handled in the proof of Theorem 7 by combining the dice of their non-separable components. Because barrier sets get precedence, we first show how to construct dice for barrier sets with Lemma 8. Once we have dice for barrier sets, we can repeatedly “peel off” the tight sets and combine their dice, which is captured in Lemma 9. We start with the existence of dice for barrier sets:

► **Lemma 8.** *Let  $\mathcal{M}$  be a non-separable matroid, and  $\mathbf{x} > \mathbf{0}$  a feasible interim rule within the winner-selection environment  $(f, \mathcal{M})$ . Let  $S$  be a barrier set for  $(f, \mathbf{x}, \mathcal{M})$ , and define  $\mathbf{x}_S$  to be  $(x(t))_{t \in S}$ . There exists a vector of distributions  $\mathcal{D} = (D_t)_{t \in S}$  over  $\mathbb{R}$ , such that  $\mathcal{D}$  implements  $\mathbf{x}_S$  in  $(f, \mathcal{M}|S)$ .*

The proof is quite technically involved, and due to space constraints, it is entirely deferred to the full version. The high-level idea is to base the dice upon any full-support continuous distribution for the dice. This distribution is scaled by different parameters  $\theta_i$  for different types  $i$ , and shifted by a constant  $\tau$ . Matching the prescribed interim winning probabilities  $x(i)$  imposes a system of non-linear equations on the  $\theta_i$ . We establish that the winning probabilities of different types satisfy certain key monotonicity properties in the parameters  $\theta_i$ ; these monotonicity properties are proved using the fact that the feasible sets form a matroid. Then, we apply the Intermediate Value Theorem inductively for all types  $i$  to non-constructively prove the existence of the desired  $\theta_i$ .

To generalize Lemma 8 to arbitrary sets of types, we will need a construction that allows us to “scale” the faces of some dice such that they will always be above/below the faces of another set of newly introduced dice; such a construction will allow us to give dice for types in barrier sets higher faces than other dice. For the types of full-support distributions over  $[0, \infty)$  we have been using so far, this would be impossible. There is a simple mapping that guarantees our desired properties: we map faces from  $(0, \infty)$  to the set  $(1, 2)$  by mapping all positive  $s \mapsto 2 - \frac{1}{1+s}$ , and mapping all negative  $s$  to  $-1$ . Notice that the new dice implement the same interim rule as the old ones: in matroid environments, the set maximizing the sum of die rolls is determined by the greedy algorithm, and hence, only the relative order between die faces matters. With the help of this mapping, we prove the following lemma, similar to Lemma 8. The proof is again only given in the full version.

► **Lemma 9.** *Let  $\mathbf{x} > \mathbf{0}$  be a feasible interim rule within a winner-selection environment  $(f, \mathcal{M})$ , where  $\mathcal{M}$  is a non-separable matroid. Fix a tight set  $S$ , and let  $\hat{S}$  be a minimal tight set that includes  $S$  as a proper subset, if such a set exists; otherwise let  $\hat{S} = T$ . Given dice  $\mathcal{D} = (D_t)_{t \in S}$  that implement  $\mathbf{x}_S$  in  $(f, \mathcal{M}|S)$ , there are dice  $\mathcal{D}' = (D'_t)_{t \in \hat{S}}$  which implement  $\mathbf{x}_{\hat{S}}$  in  $(f, \mathcal{M}|\hat{S})$ .*

**Proof of Theorem 7.** First consider the case when  $\mathcal{M}$  is non-separable. Let  $T$  be the type set with  $m$  types. We define the dice system as follows: First, for all types  $t$  with  $x(t) = 1$ , assign them a point distribution (single-sided die) at 2, so they always win. Next, for all types  $t$  with  $x(t) = 0$ , assign them a point distribution at  $-1$ , so they never win. Next, we create dice for barrier sets  $S$  according to Lemma 8. Then, starting from the barrier sets, we repeatedly apply Lemma 9 to construct dice for larger tight sets  $\hat{S} \supsetneq S$  (or all of  $T$ ) implementing  $\mathbf{x}_{\hat{S}}$ .

If  $\mathcal{M}$  is separable, let  $\mathcal{M}_1, \dots, \mathcal{M}_k$  be the components of  $\mathcal{M}$ . Using the construction from the previous paragraph for each  $\mathcal{M}_j$ , let  $\mathcal{D}_j$  be the dice set constructed for  $\mathcal{M}_j$ . Since there is no circuit containing two candidates from different components, the winner set of one component has no effect on the winner set of any other component. Thus, the union  $\bigcup_{j=1}^k \mathcal{D}_j$  of dice implements the desired interim rule. ◀



**ALGORITHM 1:** FIND BARRIER SET( $f, \mathbf{x}$ ).

---

```

1  $T^+ \leftarrow \{t \in T : f(t)x(t) > 0\}$ .
2 Define  $g(S) = f(S) - \sum_{t \in S} f(t)x(t)$  for  $S \subseteq T^+$ .
3 if  $\min \{g(S) : \emptyset \subsetneq S \subseteq T^+\} \neq 0$  then
4   return  $T^+$ .
5 else
6    $T^* \leftarrow T^+$ .
7   while there is a type  $t \in T^*$  such that  $\min \{g(S) : \emptyset \subsetneq S \subseteq T^* \setminus \{t\}\} = 0$  do  $T^* \leftarrow T^* \setminus \{t\}$ .
8   return  $T^*$ .

```

---

We use the same notation  $f(t)$  and  $x(t)$  as in Section 3. In the single-winner setting, the function  $R(S) = \mathbb{E}_{t \sim f} [r_{\mathcal{M}}(\mathbf{t} \cap S)]$  can be treated as a natural extension of  $f$  to subsets of  $T$ : given  $S \subseteq T$ , we let  $R(S) = f(S) = 1 - \prod_{i=1}^n (1 - \sum_{t \in S_i} f(t))$ , which is the probability that at least one type in  $S$  shows up. Border's constraints can then be written as follows:  $\sum_{t \in S} f(t)x(t) \leq f(S)$  for all  $S \subseteq T$ . The slack function becomes  $\sigma_{f, \mathbf{x}}(S) = f(S) - \sum_{t \in S} f(t)x(t)$  and is nonnegative everywhere when  $\mathbf{x}$  is feasible.

Tight sets and barrier sets are defined as the special case of the definition for general matroids in Section 3. The algorithm FIND BARRIER SET, given as Algorithm 1, simply implements the definition of barrier sets as minimal non-empty tight sets, and therefore correctly computes a barrier set.

The following lemma characterizes the key useful structure of barrier sets for the single-winner setting.

► **Lemma 12.** *Let  $f$  and  $\mathbf{x}$  be such that  $\mathbf{x}$  is feasible for  $f$ . If there are multiple barrier sets for  $(f, \mathbf{x})$ , then there is a candidate  $i^*$  such that each barrier set is a singleton  $\{t\}$  with  $t \in T_{i^*}$ .*

**Proof.** Let  $A, B$  be any two barrier sets. Because  $A$  and  $B$  are both tight, the lattice property of tight sets implies that  $A \cap B = \emptyset$ .

We first show that there is a candidate  $i^*$  with  $A \subseteq T_{i^*}$  and  $B \subseteq T_{i^*}$ . Suppose not for contradiction; then, there exist  $i \neq j$  and types  $t_i \in A \cap T_i$  and  $t_j \in B \cap T_j$ . With non-zero probability, the types  $t_i$  and  $t_j$  show up at the same time. However, according to the definition of a tight set, when a type in  $A$  shows up, the winner must be a candidate with type in  $A$ , and the same must hold for  $B$ . Then, the winner's type would have to be in  $A \cap B$  with non-zero probability. This contradicts the disjointness of  $A$  and  $B$ .

It remains to show that all barrier sets are singletons. Since  $A$  is tight and all types in  $A$  belong to the same candidate,  $\sum_{t \in A} f(t)x(t) = f(A) = \sum_{t \in A} f(t)$ . Hence,  $x(t) = 1$  for all  $t \in A$ , and because barrier sets are minimally tight,  $A$  must be a singleton. ◀

## 4.1 Description of the Algorithm

Given a prior  $f$  and a feasible interim rule  $\mathbf{x}$ , the recursive procedure CONSTRUCT DICE, shown in Algorithm 2, returns a family of dice  $\mathcal{D}$  implementing  $\mathbf{x}$  for  $f$ . It operates as follows. There are two simple base cases: when no candidate ever wins, and when a single type of a single candidate always wins. In the recursive case, the algorithm carefully selects a type  $t^*$  and awards its die the highest-valued face  $M^t$ . It assigns this new face a probability as large as possible, subject to still permitting implementation of  $\mathbf{x}$ . We choose  $t^*$  as a member of a barrier set; this is important in order to guarantee that the algorithm makes significant progress.



---

**ALGORITHM 2:** CONSTRUCT DICE ( $f, \mathbf{x}$ ).

---

**Input** : PDFs  $f_1, \dots, f_n$  supported on disjoint type sets  $T_1, \dots, T_n$ .  
**Input** : Interim rule  $\mathbf{x}$  feasible for  $f$ .  
**Output** : Vector of dice  $(D_t)_{t \in \biguplus_{i=1}^n T_i}$ .

- 1 Let  $T = \biguplus_i T_i$ .
- 2 Let  $T_i^+ = \{t \in T_i : f_i(t)x_i(t) > 0\}$ , and let  $T^+ = \biguplus_{i=1}^n T_i^+$ .
- 3 **if**  $T^+ = \emptyset$  **then**
- 4 **for all** types  $t \in T$ , let  $D_t$  be a single-sided die with a  $-1$  face.
- 5 **else if** there is a type  $t^* \in T^+$  with  $f(t^*)x(t^*) = 1$  **then**
- 6 Let  $D_{t^*}$  be a single-sided die with a  $+1$  face.
- 7 **for all** other types  $t \in T \setminus \{t^*\}$ , let  $D_t$  be a single-sided die with a  $-1$  face.
- 8 **else**
- 9 Let  $T^* = \text{FIND BARRIER SET}(f, \mathbf{x})$ .
- 10 Let  $t^* \in T^*$  be a type chosen arbitrarily.
- 11 Let  $(f', \mathbf{x}') = \text{DECREMENT}(f, \mathbf{x}, t^*, q^*)$ , for the largest value of  $q^* \in [0, f(t^*)x(t^*)]$  such that  $\mathbf{x}'$  is feasible for  $f'$ . /\* Note that  $f(t^*)x(t^*) < 1$ . \*/
- 12 Let  $(D'_t)_{t \in T} \leftarrow \text{CONSTRUCT DICE}(f', \mathbf{x}')$ .
- 13 Let  $M$  be the maximum possible face of any die  $D'_t$ , and  $M' := \max(M, 0) + 1$ .
- 14 Let  $D_t = D'_t$  for all types  $t \neq t^*$ .
- 15 Let  $D_{t^*}$  be the die which rolls  $M'$  with probability  $\frac{q^*}{f(t^*)}$ , and  $D'_{t^*}$  with probability  $1 - \frac{q^*}{f(t^*)}$ .
- 16 **return**  $(D_t)_{t \in T}$ .

---



---

**ALGORITHM 3:** DECREMENT( $f, \mathbf{x}, t^*, q$ ).

---

*/\*  $q \geq 0$  is the probability allocated to the highest face. Because it is a contribution to the unconditional winning probability  $f(t^*)x(t^*)$  of type  $t^*$ , and we separated out the case that a single type has unconditional winning probability 1,  $q$  satisfies  $q \leq f(t^*)x(t^*) < 1$ . \*/*

- 1 **if**  $q = f(t^*)$ , **then** let  $f'(t^*) \leftarrow 0$  and  $x'(t^*) \leftarrow 0$
- 2 **else** let  $f'(t^*) \leftarrow \frac{f(t^*)-q}{1-q}$  and  $x'(t^*) \leftarrow \frac{f(t^*)x(t^*)-q}{f(t^*)-q}$ .
- 3 Let  $i^*$  be such that  $t^* \in T_{i^*}$ .
- 4 **for all**  $t \in T_{i^*}, t \neq t^*$ , let  $f'(t) \leftarrow \frac{f(t)}{1-q}$  and  $x'(t) \leftarrow x(t)$ .
- 5 **for all**  $t \in T \setminus T_{i^*}$ , let  $f'(t) \leftarrow f(t)$  and  $x'(t) \leftarrow \frac{x(t)}{1-q}$ .
- 6 **return**  $(f', \mathbf{x}')$ .

---

The subroutine DECREMENT, shown as Algorithm 3, essentially conditions both  $f$  and  $\mathbf{x}$  on the face  $M'$  not winning. Specifically, DECREMENT computes the conditional type distribution  $f'$ , and an interim rule  $\mathbf{x}'$ , such that if there were a dice implementation of  $\mathbf{x}'$  for  $f'$ , then adding  $M'$  to the die of  $t^*$  would yield a set of dice implementing  $\mathbf{x}$  for  $f$ .

We now provide a formal analysis of our algorithm. Theorem 11 follows from Lemmas 13–15.

► **Lemma 13.** *If CONSTRUCT DICE terminates, it outputs dice implementing  $\mathbf{x}$  for  $f$ .*

► **Lemma 14.** *CONSTRUCT DICE terminates after at most  $m^2$  recursive calls. (Recall that  $m = \sum_i |T_i|$ .)*

► **Lemma 15.** *Excluding the recursive call, each invocation of CONSTRUCT DICE can be implemented in time polynomial in  $n$  and  $m$ .*

## 4.2 Proof of Lemma 13 (Correctness)

We prove the lemma by induction over the algorithm's calls. Correctness is obvious for the two base cases: when  $T^+ = \emptyset$  (no type should win), and when there exists a type  $t^*$  with  $f(t^*)x(t^*) = 1$  ( $t^*$  always shows up and should always win). For the inductive step, suppose that the recursive call in step 12 returns dice  $\mathcal{D}' = (D'_t)_{t \in T}$ , correctly implementing  $\mathbf{x}'$  for  $f'$ , and let  $\mathcal{D} = (D_t)_{t \in T}$  be the new dice defined in steps 14 and 15.

We analyze the interim winning probability of each type when using the dice-based winner selection rule given by  $\mathcal{D}$ . For each type  $t$ , let  $\bar{v}_t \sim D_t$  be a roll of the die for type  $t$ , and for each  $i$ , let  $t_i \sim f_i$  be a draw of a type; all  $\bar{v}_t$  and  $t_i$  are mutually independent. In other words, we may assume that the die of *every* type is rolled (including types that do not show up), then the type profile is drawn independently. The winning type is then the type  $t_i$  with largest positive  $\bar{v}_{t_i}$ ; if all  $\bar{v}_{t_i}$  are negative, then no type wins.

Let  $t^* \in T_{i^*}$  be as defined in step 10. Let  $\mathcal{E}$  be the event that  $i^*$  has type  $t^*$  and that  $\bar{v}_{t^*} = M'$ , and let  $\bar{\mathcal{E}}$  be its complement. By independence of the random choices, the probability of  $\mathcal{E}$  is  $f(t^*) \cdot \frac{q^*}{f(t^*)} = q^*$ . Type  $t^*$  always wins under the event  $\mathcal{E}$ . Conditioned on  $\bar{\mathcal{E}}$ , each  $\bar{v}_t$  (including  $\bar{v}_{t^*}$ ) is distributed as a draw from  $D'_t$ , the type vector  $\mathbf{t}$  is distributed as a draw from  $f'_1 \times \dots \times f'_n$ , and the  $\bar{v}_t$ 's and  $\mathbf{t}$  are mutually independent. By the inductive hypothesis, conditioned on  $\bar{\mathcal{E}}$ , each type  $t$  wins with probability  $f'(t)x'(t)$ . Using the definition of  $f'$  and  $\mathbf{x}'$  from the DECREMENT subroutine, the total winning probability for  $t^*$  is  $q^* \cdot 1 + (1 - q^*) \cdot f'(t^*)x'(t^*) = q^* + (1 - q^*) \cdot \frac{f(t^*)x(t^*) - q^*}{1 - q^*} = f(t^*)x(t^*)$ . For  $t \neq t^*$ , the total winning probability is  $q^* \cdot 0 + (1 - q^*)f'(t)x'(t) = (1 - q^*) \cdot \frac{f(t)x(t)}{1 - q^*} = f(t)x(t)$ . Therefore, the interim winning probability for each type  $t$  is  $x(t)$ , and the dice  $\mathcal{D}$  implement  $\mathbf{x}$  for  $f$ .

## 4.3 Proof of Lemma 14 (Number of Recursive Calls)

The following lemma is essential in that it shows that invoking DECREMENT maintains feasibility and tightness of sets.

► **Lemma 16.** *Let  $f, \mathbf{x}, t^*$  and  $q$  be valid inputs for DECREMENT, and  $f', \mathbf{x}'$  the output of the call to DECREMENT( $f, \mathbf{x}, t^*, q$ ). Let  $S$  be any set of types with  $t^* \in S$ . Then,*

1. *The Border constraint for  $S$  is satisfied for  $(f', \mathbf{x}')$  if and only if it is satisfied for  $(f, \mathbf{x})$ .*
2. *The Border constraint for  $S$  is tight for  $(f', \mathbf{x}')$  if and only if it is tight for  $(f, \mathbf{x})$ .*

**Proof.** We will show that the slack for every  $S \ni t^*$  satisfies  $\sigma_{f', \mathbf{x}'}(S) = \frac{\sigma_{f, \mathbf{x}}(S)}{1 - q}$ , which implies both claims. Let  $i^*$  be such that  $t^* \in T_{i^*}$ . Using the definitions of  $f'$  and  $\mathbf{x}'$ ,

$$\begin{aligned} \sigma_{f', \mathbf{x}'}(S) &= f'(S) - \sum_{t \in S} f'(t)x'(t) \\ &= 1 - \left(1 - \frac{f(S_{i^*}) - q}{1 - q}\right) \cdot \prod_{i \neq i^*} (1 - f(S_i)) - \frac{(\sum_{t \in S} f(t)x(t)) - q}{1 - q} \\ &= \frac{1}{1 - q} \cdot \left(1 - (1 - f(S_{i^*})) \cdot \prod_{i \neq i^*} (1 - f(S_i)) - \sum_{t \in S} f(t)x(t)\right) = \frac{\sigma_{f, \mathbf{x}}(S)}{1 - q}. \quad \blacktriangleleft \end{aligned}$$

We are now ready to prove Lemma 14. We will show that with each recursive invocation of CONSTRUCT DICE (step 12), at least one of the following happens: (1) The number of active types  $|T^+|$  decreases; (2) The size of the barrier set  $|T^*|$  decreases.

Notice that the number of active types or the size of a barrier set never *increase*. Because the size of the barrier set can only decrease at most  $m$  times, the number of active types must decrease at least every  $m$  recursive invocations. It, too, can decrease at most  $m$  times, implying the claim of the lemma.

Let  $T^*$ ,  $t^*$ , and  $(q^*, f', \mathbf{x}')$  be as chosen in steps 9, 10, and 11, respectively. Let candidate  $i^*$  be such that  $t^* \in T_{i^*}$ . If  $q^* = f(t^*)x(t^*)$ , then the type  $t^*$  will be inactive in  $(f', \mathbf{x}')$ , and there will be one fewer active type in the subsequent invocation of **CONSTRUCT DICE** (step 12). We distinguish the cases  $|T^*| = 1$  and  $|T^*| > 1$ .

If  $|T^*| = 1$ , then  $T^* = \{t^*\}$ . By definition of a barrier set,  $T^*$  is tight, implying (for a singleton set) that  $x(t^*) = 1$ . We claim that  $q^*$  is set to  $f(t^*) = f(t^*)x(t^*)$  in step 11, implying that the number of active types decreases. To prove that  $q^* = f(t^*)$ , we will show that this choice of  $q^*$  is feasible in the invocation of **DECREMENT** $(f, \mathbf{x}, t^*, f(t^*))$ . Consider the  $\hat{f}, \hat{\mathbf{x}}$  resulting from such an invocation of **DECREMENT**. Lemma 16 implies that the feasibility of each Border constraint corresponding to a set  $S \ni t^*$  is preserved for  $(\hat{f}, \hat{\mathbf{x}})$ . For type sets  $S$  excluding  $t^*$ ,

$$\begin{aligned} \sigma_{\hat{f}, \hat{\mathbf{x}}}(S) &= \hat{f}(S) - \sum_{t \in S} \hat{f}(t) \hat{\mathbf{x}}(t) = 1 - \left(1 - \frac{f(S_{i^*})}{1 - f(t^*)}\right) \cdot \prod_{i \neq i^*} (1 - f(S_i)) - \frac{\sum_{t \in S} f(t)x(t)}{1 - f(t^*)} \\ &= \frac{1}{1 - f(t^*)} \cdot \left(1 - (1 - f(S_{i^*}) + f(t^*)) \cdot \prod_{i \neq i^*} (1 - f(S_i))\right. \\ &\quad \left. - \left(\sum_{t \in S} f(t)x(t) + f(t^*)x(t^*)\right)\right) \\ &= \frac{\sigma_{f, \mathbf{x}}(S \cup \{t^*\})}{1 - f(t^*)}, \end{aligned}$$

which is nonnegative because  $\mathbf{x}$  is feasible for  $f$ . Therefore, step 11 indeed chooses  $q^* = f(t^*)$ .

Next, we consider the case  $|T^*| > 1$ , and assume that  $q^* < f(t^*)x(t^*)$  (since otherwise, we are done). Then, the set of active types  $T^+$  is the same for both  $(f, \mathbf{x})$  and  $(f', \mathbf{x}')$ .

If the instance  $(f', \mathbf{x}')$  for the recursive call has multiple barrier sets, then by Lemma 12, they are all singletons, and indeed the size of the barrier set in the next recursive call (which is 1) is strictly smaller than  $|T^*|$ . So we assume that  $(f', \mathbf{x}')$  has a unique barrier set  $T'$ .

Because  $|T^*| > 1$ , Lemma 12 implies that  $T^*$  is the unique barrier set for  $(f, \mathbf{x})$ . Therefore, by the definition of barrier sets,  $T^*$  (and hence also  $t^*$ ) is contained in every tight set of active types for  $(f, \mathbf{x})$  (if any). Because  $t^*$  is contained in all tight sets, Lemma 16 implies that for every  $q \in [0, f(t^*)x(t^*)]$ , the result of **DECREMENT** $(f, \mathbf{x}, t^*, q)$  does not violate any constraints which are already tight for  $(f, \mathbf{x})$ , and in fact preserves their tightness.

Because all other constraints have slack, the optimal  $q^*$  is strictly positive. By assumption, we also have that  $q^* < f(t^*)x(t^*)$ ; therefore, a non-empty set  $S'$  which was not tight for  $(f, \mathbf{x})$  must have become tight for  $(f', \mathbf{x}') = \text{DECREMENT}(f, \mathbf{x}, t^*, q^*)$ . By Lemma 16, this set  $S'$  does not include  $t^*$ . Because discarding inactive types preserves tightness, we may assume without loss of generality that  $S' \subseteq T^+$ .

We have shown the existence of a non-empty tight set  $S' \subseteq T^+ \setminus \{t^*\}$  for  $(f', \mathbf{x}')$ . By definition, the barrier set  $T'$  is the (unique, in our case) minimal tight set for  $(f', \mathbf{x}')$ , so  $T' \subseteq S'$ . We distinguish two cases, based on the possible definitions of barrier sets:

- If  $T^* = T^+$ , then because  $S' \subsetneq T^+ = T^*$ , the barrier set  $T'$  is strictly smaller than  $T^*$ .
- If  $T^*$  is tight for  $(f, \mathbf{x})$ , then it is also tight for  $(f', \mathbf{x}')$  by Lemma 16; since both  $S'$  and  $T^*$  are tight for  $(f', \mathbf{x}')$ , we get that  $T' \subseteq T^* \cap S' \subseteq T^* \setminus \{t^*\}$  is strictly smaller than  $T^*$ .

#### 4.4 Proof of Lemma 15 (Runtime per Call)

There are only two steps for which polynomial runtime is not immediate: the computation of  $q^*$  in step (11) of **CONSTRUCT DICE**, and finding a non-empty minimizer of a submodular function in steps (3) and (7) of **FIND BARRIER SET**. We prove polynomial-time implementability of both steps in the following lemmas.

► **Lemma 17.** *In step 11 of **CONSTRUCT DICE**,  $q^*$  can be computed in  $\text{poly}(m)$  time.*

**Proof.** Let  $f$ ,  $\mathbf{x}$ , and  $t^*$  be as in step 11, and let the candidate  $i^*$  be such that  $t^* \in T_{i^*}$ . For each  $q \in [0, f(t^*)x(t^*)] \subseteq [0, 1)$ , let  $(f_q, \mathbf{x}_q) = \text{DECREMENT}(f, \mathbf{x}, t^*, q)$  be the result of running **DECREMENT** $(f, \mathbf{x}, t^*, q)$  with parameter  $q$ . Lemma 16 implies that all Border constraints for  $S \ni t^*$  remain feasible for  $(f_q, \mathbf{x}_q)$ . For type sets  $S \not\ni t^*$ , we can write the slack in the corresponding Border constraint as a function of  $q$  as follows:

$$\begin{aligned} \sigma_{f_q, \mathbf{x}_q}(S) &= f_q(S) - \sum_{t \in S} f_q(t)x_q(t) \\ &= 1 - \left(1 - \frac{f(S_{i^*})}{1-q}\right) \cdot \prod_{i \neq i^*} (1 - f(S_i)) - \frac{\sum_{t \in S} f(t)x(t)}{1-q} \\ &= \frac{1}{1-q} \cdot \left(1 - (1 - f(S_{i^*}) - q) \cdot \prod_{i \neq i^*} (1 - f(S_i)) - \sum_{t \in S} f(t)x(t) - q\right) \\ &= \frac{1}{1-q} (\sigma_{f, \mathbf{x}}(S) - qf(S \setminus S_{i^*})). \end{aligned}$$

The preceding expression is nonnegative if and only if  $q \leq h(S) := \frac{\sigma_{f, \mathbf{x}}(S)}{f(S \setminus S_{i^*})}$ . Therefore,  $q^*$  is the minimum of  $f(t^*)x(t^*)$  and  $\min_{S \subseteq T \setminus \{t^*\}} h(S)$ . The function  $h(S)$  does not appear to be submodular, and hence efficient minimization is not immediate. We utilize Theorem 4 to reduce the search space and compute  $q^*$  efficiently.

For an interim rule  $\mathbf{x} : T \rightarrow [0, 1]$ , we call  $S \subseteq T$  a *level set* of  $\mathbf{x}$  if there exists an  $\alpha \in [0, 1]$  such that  $S = \{t \in T : x(t) > \alpha\}$ . If  $q^* = \min_{S \not\ni t^*} h(S)$ , then at least one Border constraint just becomes tight at  $q = q^*$ . Theorem 4 implies that at least one level set of  $\mathbf{x}_{q^*}$  corresponds to one of these newly tightened constraints, and  $h$  is minimized by such a level set. It follows that, in order to compute  $q^*$ , it suffices to minimize  $h$  over all those sets  $S \not\ni t^*$  which could possibly arise as level sets of some  $\mathbf{x}_q$  for  $q \in [0, f(t^*)x(t^*)]$ .

Let  $t_1, \dots, t_K$  be the types in  $T_{i^*} \setminus \{t^*\}$ , ordered by non-increasing  $x(t)$ ; similarly, let  $t'_1, \dots, t'_L$  be the types in  $T \setminus T_{i^*}$ , ordered by non-increasing  $x(t)$ . The relative order of types in  $T_{i^*} \setminus \{t^*\}$  is the same under  $x_q(t)$  as under  $x(t)$ , because  $x_q(t) = x(t)$ ; similarly, the relative order of types in  $T \setminus T_{i^*}$  is the same under  $x_q(t)$  as under  $x(t)$ , because  $x_q(t) = \frac{x(t)}{1-q}$ . Therefore, the family  $\{\{t_1, \dots, t_k, t'_1, \dots, t'_\ell\} : k \leq K, \ell \leq L\}$  includes all level sets of every  $\mathbf{x}_q$  excluding  $t^*$ . There are at most  $m^2$  type sets in this family, and those sets can be enumerated efficiently to minimize  $h$ . ◀

► **Lemma 18.** *There is an algorithm for computing a non-empty minimizer of a submodular function in the value oracle model, with runtime polynomial in the size of the ground set.*

**Proof.** For a submodular function  $g : 2^T \rightarrow \mathbb{R}$ , let  $g_t(S) = g(S \cup \{t\})$ , for  $t \in T$ .  $g_t$  is also submodular, and can be minimized in time polynomial in  $|T|$  [10]. Let  $S_t$  be a minimizer of  $g_t$  and  $t^* \in \arg \min_{t \in T} g_t(S_t)$ .  $S_{t^*} \cup \{t^*\}$  is a non-empty minimizer of  $g$ . ◀

## 5 From Winner-Selecting Dice to Interim Rules for Single-Winner Settings

Having shown how to compute winner-selecting dice implementing a given interim rule, we next show the easier converse direction: how to compute the interim rule given winner-selecting dice  $\{D_{i,t}\}$  in single-winner environments. As before, we denote the type set of candidate  $i$  by  $T_i$ , and assume without loss of generality that the type sets of different candidates are disjoint. For simplicity of exposition, we assume that each die has a given finite support<sup>4</sup>, and we write  $U_i := \bigcup_{t \in T_i} \text{supp}(D_{i,t})$  for the combined support of candidate  $i$ 's dice. We also assume that we can evaluate the probability  $\Pr[D_{i,t} = u]$  of the face labeled with  $u$ , for all candidates  $i$ , types  $t$ , and faces  $u \in U_i$ .

► **Theorem 19.** *Consider a single-winner selection environment with  $n$  candidates and independent priors  $f_1, \dots, f_n$ , where  $f_i$  is supported on  $T_i$ . Given dice  $\{D_{i,t} \mid i \in [n], t \in T_i\}$  represented explicitly, the interim rule of the corresponding dice-based winner selection rule can be computed in time polynomial in  $n$ ,  $m = \sum_i |T_i|$ , and the total support size  $|\bigcup_i U_i|$  of all the dice.*

**Proof.** First, we can compute the probability mass function of each candidate  $i$ 's (random) score  $\bar{v}_i$ . Given  $u \in U_i$ , we have  $\Pr[\bar{v}_i = u] = \sum_{t \in T_i} f_i(t) \cdot \Pr[D_{i,t} = u]$ .

From this probability mass function, we easily compute  $\Pr[\bar{v}_i \leq u]$  for each  $u \in U_i$  by the appropriate summation. When all dice faces are distinct, this is all we need; since  $\Pr[\bar{v}_{i'} < u] = \Pr[\bar{v}_{i'} \leq u]$  for  $i' \neq i$  and  $u \in U_i$ , the interim rule is given by the following simple equation:

$$x_i(t) = \sum_{u \in \text{supp}(D_{i,t})} \Pr[D_{i,t} = u] \cdot \prod_{i' \neq i} \Pr[\bar{v}_{i'} \leq u].$$

When the dice's faces are not distinct, recall that we break ties uniformly at random. To account for the contribution of this tie-breaking rule, we need the distribution of the number of other candidates that tie candidate  $i$ 's score of  $u$ ; this is a Poisson Binomial distribution. More, precisely, we need the Poisson Binomial distribution with the  $n - 1$  parameters  $\left( \frac{\Pr[\bar{v}_{i'} = u]}{\Pr[\bar{v}_{i'} \leq u]} \right)_{i' \neq i}$ ; we denote its probability mass function by  $B_{i,u}$ . It is well known, and easy to verify, that a simple dynamic program computes the probability mass function of a Poisson Binomial distribution in time polynomial in its number of parameters. Therefore, we can compute  $B_{i,u}(k)$  for each  $i \in [n]$ ,  $u \in U_i$  and  $k \in \{1, \dots, n - 1\}$ . The interim rule is then given by the following equation:

$$x_i(t) = \sum_{u \in \text{supp}(D_{i,t})} \Pr[D_{i,t} = u] \cdot \left( \prod_{i' \neq i} \Pr[\bar{v}_{i'} \leq u] \right) \cdot \sum_{k=0}^{n-1} \frac{B_{i,u}(k)}{k+1}.$$

It is easy to verify that all the above computations satisfy the claimed runtime. ◀

<sup>4</sup> Our approach extends easily to the case of continuously supported dice, so long as we can perform integration with respect to the distributions of the various dice.

## 6 Winner-Selecting Dice and Persuasion

In this section, we investigate the existence of winner-selecting dice for instances of Bayesian persuasion. Our main result (Theorem 20) is to exhibit an instance with independent non-identical actions for which there is no optimal signaling scheme that can be implemented using winner-selecting dice. This result is contrasted with Theorem 21, which shows that when the actions' types are not just independent, but identically distributed as well, a dice-based implementation always *does* exist.

► **Theorem 20.** *There is an instance of Bayesian persuasion (given in Table 1) with independent actions which does not admit a dice-based implementation of any optimal signaling scheme. Consequently, there exists a second-order interim rule which does not admit a dice-based implementation.*

► **Theorem 21.** *Every Bayesian persuasion instance with i.i.d. actions admits an optimal dice-based signaling scheme. Moreover, when the prior type distribution is given explicitly, the corresponding dice can be computed in time polynomial in the number of actions and types.*

The negative result of Theorem 20 has interesting implications. Since second-order interim rules summarize all the attributes of a winner selection rule relevant to persuasion, second-order interim rules, unlike their first-order brethren, can in general not be implemented by dice. Most importantly, this result draws a sharp contrast between persuasion and single-item auctions, despite their superficial similarity: it rules out a Myerson-like virtual-value characterization of optimal persuasion schemes, and it joins the #P-hardness result of [8] as evidence of the intractability of optimal persuasion.

### 6.1 Basics of Bayesian Persuasion

In *Bayesian persuasion*, the  $n$  candidates are *actions* which a *receiver* can take. Each action  $i$  has a type  $t_i$ , drawn *independently*<sup>5</sup> from the set  $T_i$ , according to a commonly known distribution  $f_i$ . Each type  $t_i$  has associated payoffs  $s(i, t_i)$  and  $r(i, t_i)$  for the sender and receiver, respectively. The *sender* (or *principal*) also has access to the *actual* draws  $\mathbf{t} = (t_1, \dots, t_n)$  of the types, and would like to use this leverage to persuade the receiver to take an action favorable to him<sup>6</sup>.

Thereto, the sender can commit to a (typically randomized) policy  $\mathcal{A}$ — called a *signaling scheme*— of revealing some of this information to the receiver. It was shown by Kamenica and Gentzkow [12] that the sender can restrict attention, without loss, to *direct schemes*: randomized functions  $\mathcal{A}$  mapping type profiles to recommended actions. Naturally, the function must be *persuasive*: if action  $i$  is recommended, the receiver's posterior expected utility from action  $i$  must be no less than her posterior expected utility from any other action  $i'$ . In this sense, direct schemes can be viewed as winner selection rules in which the actions are the candidates, and persuasiveness constraints must be obeyed.

### 6.2 Proof of Theorems

**Proof of Theorem 20.** The persuasion instance, shown in Table 1, features three actions  $\{A, B, C\}$ , each of which has two types  $\{1, 2\}$ . The types of the different actions are distributed independently. In the instance, the sender's utility from any particular action is a constant, independent of the action's type.

<sup>5</sup> The draws are independent in this paper. In more general Bayesian persuasion models, they can be correlated.

<sup>6</sup> To avoid ambiguities, we always use male pronouns for the sender and female ones for the receiver.

■ **Table 1** A Persuasion instance with no dice-based implementation. The notation  $p \times (s, r)$  denotes that the type  $(s, r)$  (in which the sender and receiver payoffs are  $s$  and  $r$ , respectively) has probability  $p$ .

action \ type	1	2
A	$0.5 \times (100, 2)$	$0.5 \times (100, -\infty)$
B	$0.99 \times (1, 3)$	$0.01 \times (1, -\infty)$
C	$0.5 \times (0, 0)$	$0.5 \times (0, 6)$

One (optimal, as we will show implicitly) signaling scheme is the following. (In writing a type vector, here and below, we use  $*$  to denote that the type of an action is irrelevant.)

- If the type vector is  $(1, *, 1)$ , then recommend action  $A$ .
- If the type vector is  $(1, *, 2)$ , then recommend each of  $A, C$  with equal probability  $\frac{1}{2}$ .
- If the type vector is  $(2, 1, *)$ , then recommend action  $B$ .
- If the type vector is  $(2, 2, *)$ , then recommend action  $C$ .

While this is not the unique optimal scheme, we next prove that none of the optimal persuasion schemes admit a dice-based implementation.

The given signaling scheme recommends action  $A$  with probability  $3/8$  overall, action  $B$  with probability  $\frac{99}{200}$  overall, and action  $C$  with the remaining probability. No persuasive signaling scheme can recommend  $A$  with probability strictly more than  $3/8$ , because conditioned on receiving the recommendation  $A$ , action  $C$  must be at least twice as likely to be of type 1 as of type 2, in addition to action  $A$  being of type 1 with probability 1. Similarly, no persuasive scheme can recommend  $B$  with probability strictly more than  $\frac{99}{200}$ , because action  $C$  must be at least as likely to be of type 1 as of type 2 when  $C$  is recommended, in addition to action  $B$  being of type 1 with probability 1. Hence, any optimal signaling scheme must recommend  $A$  with probability  $3/8$  and  $B$  with probability  $\frac{99}{200}$ , and the given scheme is in fact optimal.

Suppose for a contradiction that there exist dice  $(D_{i,j})_{i \in \{A,B,C\}, j \in \{1,2\}}$  implementing an optimal signaling scheme. We gradually derive properties of these optimal signaling schemes, eventually leading to a contradiction.

1. Since action  $A$  can never be recommended when it has type 2 (the receiver would never follow the recommendation), it must be recommended with probability  $\frac{3}{4}$  conditioned on having type 1.
2. In particular, whenever the type profile is  $(1, *, 1)$ , action  $A$  must be recommended, regardless of the type of action  $B$ . This is because action  $C$  must be at least twice as likely of type 1 as of type 2 for a recommendation of  $A$  to be persuasive.
3. Therefore, all faces on  $D_{A,1}$  must be larger than all faces on  $D_{B,1}$  and on  $D_{C,1}$ .
4. Because of this, action  $B$  can never be recommended when the type profile is  $(1, *, 2)$ .
5. Thus, when the type profile is  $(1, *, 2)$ , the signaling scheme has to recommend each of  $A$  and  $C$  with probability  $\frac{1}{2}$ . (The recommendation could of course follow different distributions based on the type of  $B$ ; such a correlation is immaterial for our argument.)
6. Given that action  $B$  cannot be recommended when action  $A$  has type 1, or when action  $B$  has type 2, it must *always* be recommended for type vectors  $(2, 1, *)$ .
7. This implies that all faces of  $D_{B,1}$  must be larger than all faces on  $D_{C,1}$  and on  $D_{C,2}$ .
8. This is a contradiction to Step 5, which states that with positive probability,  $D_{C,2}$  beats  $D_{B,1}$ .



Thus, we have proved that there is no dice-based implementation of any optimal signaling scheme for the given instance. ◀

**Proof of Theorem 21.** When the actions' (or more generally: candidates') type distributions are i.i.d., i.e.,  $T_i$  and  $f_i$  are the same for all candidates  $i$ , Dughmi and Xu [8] have shown that there is an optimal *symmetric* signaling scheme, or more generally a symmetric second-order interim rule  $\mathbf{X}$ . We show that any symmetric second-order interim rule  $\mathbf{X}$  is uniquely determined by its first-order component, a fact implicit in [8].

For symmetric rules,  $x_{i,i',t}$  depends only on whether  $i = i'$  or  $i \neq i'$ , but not on the identities of the candidates  $i$  and  $i'$ . Therefore,  $\mathbf{X}$  can be equivalently described by two type-indexed vectors  $\mathbf{y}$  and  $\mathbf{z}$ , where  $y_t = x_{i,i,t}$  for all candidates  $i$ , and  $z_t = x_{i,i',t}$  for all candidates  $i$  and  $i'$  with  $i \neq i'$ . The vector  $\mathbf{y}$  is a first-order interim rule, and we refer to it as the *first-order component* of  $\mathbf{X}$ . If  $\mathbf{X}$  is feasible and implemented by  $\mathcal{A}$ , then  $y_t = x_{i,i,t} = \Pr[\mathcal{A}(t) = i \mid t_i = t]$  for all candidates  $i$ , so  $\mathbf{y}$  is the first-order interim rule implemented by  $\mathcal{A}$ . For every candidate  $i$  and type  $t$ , we have

$$1 = \sum_{i'=1}^n \Pr[\mathcal{A}(t) = i' \mid t_i = t] = (n-1)z_t + y_t.$$

Therefore,  $\mathbf{z} = \frac{1-\mathbf{y}}{n-1}$ , and the first-order component of a symmetric second-order interim rule suffices to fully describe it. The second-order rule is also, by the preceding argument, efficiently computable from its first-order component, and is feasible if and only if its first-order component is a feasible symmetric interim rule. Moreover, by [5], feasibility of symmetric second-order interim rules can be checked in time polynomial in the number of types and candidates, and given a feasible symmetric second-order interim rule  $\mathbf{X}$ , a winner selection rule implementing  $\mathbf{X}$  can be evaluated in polynomial time. ◀

## 7 Directions for Future Work

We have begun an investigation of dice-based winner selection rules, in which each of several candidates independently draws a “score” from a distribution (rolling a “die”), and a candidate set is selected to maximize the sum of scores, subject to a feasibility constraint. We have shown that dice-based winner selection rules can implement all first-order interim rules with matroid constraints, but not all second-order interim rules; in particular, there are instances of Bayesian persuasion in which no optimal signaling scheme can be implemented using dice.

A natural direction for future work is to understand the limits of dice-based winner selection rules. While our existence proof uses matroid properties, matroid constraints are not the limit of implementability by dice: in the full version, we show an example in which the feasible sets do not form a matroid, yet every feasible interim rule within the environment is implementable with dice. This rules out a characterization of the form “a feasibility constraint  $\mathcal{I}$  has all feasible  $(\mathbf{x}, f)$  implementable by dice if and only if  $\mathcal{I}$  is a matroid.” In fact, we do not know of any feasibility constraint and corresponding first-order feasible interim rule for which a dice-based implementation can be ruled out, though we strongly suspect that such examples exist. A difficulty in verifying our conjecture is that we are not aware of a useful general technique for proving the *non-existence* of a dice-based implementation for a given interim rule.

Another direction is to find an efficient algorithm for matroid environments. To derive an efficient algorithm from our existential proof, the functions  $g_t$  and  $h_t$  would have to be evaluated efficiently. Furthermore, even if a set of continuous dice is given, it is still unclear how to convert them to dice with finitely many faces in polynomial time.

## References

- 1 Nibia Aires, Johan Jonasson, and Olle Nerman. Order sampling design with prescribed inclusion probabilities. *Scandinavian journal of statistics*, 29(1):183–187, 2002.
- 2 Saeed Alaei, Hu Fu, Nima Haghpanah, Jason D. Hartline, and Azarakhsh Malekian. Bayesian optimal auctions via multi-to single-agent reduction. In *Proc. 13th ACM Conf. on Electronic Commerce*, page 17, 2012.
- 3 Kim C. Border. Implementation of Reduced Form Auctions: A Geometric Approach. *Econometrica*, 59(4):1175–1187, 1991.
- 4 Kim C. Border. Reduced form auctions revisited. *Economic Theory*, 31(1):167–181, 2007.
- 5 Yang Cai, Constantinos Daskalakis, and S. Matthew Weinberg. An algorithmic characterization of multi-dimensional mechanisms. In *Proc. 44th ACM Symp. on Theory of Computing*, pages 459–478, 2012.
- 6 Yang Cai, Constantinos Daskalakis, and S. Matthew Weinberg. Optimal multi-dimensional mechanism design: Reducing revenue to welfare maximization. In *Proc. 53rd IEEE Symp. on Foundations of Computer Science*, pages 130–139, 2012.
- 7 Shaddin Dughmi. Algorithmic information structure design: a survey. *ACM SIGecom Exchanges*, 15(2):2–24, 2017.
- 8 Shaddin Dughmi and Haifeng Xu. Algorithmic Bayesian Persuasion. In *Proc. 48th ACM Symp. on Theory of Computing*, pages 412–425, 2016.
- 9 Parikshit Gopalan, Noam Nisan, and Tim Roughgarden. Public projects, Boolean functions and the borders of Border’s Theorem. In *Proc. 16th ACM Conf. on Economics and Computation*, page 395, 2015.
- 10 Martin Grötschel, László Lovász, and Alexander Schrijver. *Geometric algorithms and combinatorial optimization*, volume 2. Springer Science & Business Media, 2012.
- 11 Jason D. Hartline. *Mechanism design and approximation*. Now Publishers, 2013.
- 12 Emir Kamenica and Matthew Gentzkow. Bayesian persuasion. *American Economic Review*, 101(6):2590–2615, 2011.
- 13 Jean-Bernard Lasserre. *Moments, positive polynomials and their applications*, volume 1. World Scientific, 2010.
- 14 Eric Maskin and John Riley. Optimal auctions with risk averse buyers. *Econometrica*, 52(6):1473–1518, 1984.
- 15 Steven A. Matthews. On the implementability of reduced form auctions. *Econometrica*, 52(6):1519–1522, 1984.
- 16 Konrad Mierendorff. Asymmetric reduced form auctions. *Economics Letters*, 110(1):41–44, 2011.
- 17 Roger B. Myerson. Optimal Auction Design. *Mathematics of Operations Research*, 6(1):58–73, 1981.
- 18 James G Oxley. *Matroid theory*, volume 3. Oxford University Press, USA, 2006.
- 19 Mallesh M. Pai and Rakesh Vohra. Optimal auctions with financially constrained buyers. *Journal of Economic Theory*, 150:383–425, 2014.
- 20 Bengt Rosén. Asymptotic theory for order sampling. *Journal of Statistical Planning and Inference*, 62(2):135–158, 1997.
- 21 Hassler Whitney. On the abstract properties of linear dependence. *American Journal of Mathematics*, 57(3):509–533, 1935.

# Spanoids – An Abstraction of Spanning Structures, and a Barrier for LCCs

Zeev Dvir<sup>1</sup>

Dept of Computer Science and Dept of Mathematics, Princeton University, Princeton, NJ, USA  
zeev.dvir@gmail.com

Sivakanth Gopi<sup>2</sup>

Microsoft Research, Redmond, WA, USA  
sigopi@microsoft.com

Yuzhou Gu<sup>3</sup>

MIT, Cambridge, MA, USA  
yuzhougu@mit.edu

Avi Wigderson<sup>4</sup>

Institute of Advanced Study, Princeton, NJ, USA  
avi@math.ias.edu

---

## Abstract

---

We introduce a simple logical inference structure we call a *spanoid* (generalizing the notion of a matroid), which captures well-studied problems in several areas. These include combinatorial geometry (point-line incidences), algebra (arrangements of hypersurfaces and ideals), statistical physics (bootstrap percolation), network theory (gossip / infection processes) and coding theory. We initiate a thorough investigation of spanoids, from computational and structural viewpoints, focusing on parameters relevant to the applications areas above and, in particular, to questions regarding Locally Correctable Codes (LCCs).

One central parameter we study is the *rank* of a spanoid, extending the rank of a matroid and related to the dimension of codes. This leads to one main application of our work, establishing the first known barrier to improving the nearly 20-year old bound of Katz-Trevisan (KT) on the dimension of LCCs. On the one hand, we prove that the KT bound (and its more recent refinements) holds for the much more general setting of spanoid rank. On the other hand we show that there exist (random) spanoids whose rank matches these bounds. Thus, to significantly improve the known bounds one must step out of the spanoid framework.

Another parameter we explore is the *functional rank* of a spanoid, which captures the possibility of turning a given spanoid into an actual code. The question of the relationship between rank and functional rank is one of the main questions we raise as it may reveal new avenues for constructing new LCCs (perhaps even matching the KT bound). As a first step, we develop an entropy relaxation of functional rank to create a small constant gap and amplify it by tensoring to construct a spanoid whose functional rank is smaller than rank by a polynomial factor. This is evidence that the entropy method we develop can prove polynomially better bounds than KT-type methods on the dimension of LCCs.

To facilitate the above results we also develop some basic structural results on spanoids including an equivalent formulation of spanoids as set systems and properties of spanoid products. We feel that given these initial findings and their motivations, the abstract study of spanoids merits further investigation. We leave plenty of concrete open problems and directions.

---

<sup>1</sup> Research supported by NSF CAREER award DMS-1451191 and NSF grant CCF-1523816.

<sup>2</sup> Research supported by NSF CAREER award DMS-1451191 and NSF grant CCF-1523816.

<sup>3</sup> Research supported by Jacobs Family Presidential Fellowship.

<sup>4</sup> Research was partially supported by NSF grant CCF-1412958.



**2012 ACM Subject Classification** Theory of computation → Error-correcting codes

**Keywords and phrases** Locally correctable codes, spanoids, entropy, bootstrap percolation, gossip spreading, matroid, union-closed family

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2019.32

**Related Version** Full version at <https://arxiv.org/abs/1809.10372>.

**Acknowledgements** The second author would like to thank Sumegha Garg for helpful discussions. The third author would like to thank Yuri Polyanskiy for helpful discussions.

## 1 Introduction

This (somewhat long) introduction will be organized as follows. We begin by discussing Locally Correctable Codes (LCCs) and the main challenges they present as this was the primary motivation for this work. We proceed to define spanoids as an abstraction of LCCs, and state some results about their rank which hopefully illuminate the difficulties with LCCs in a new light. We continue by describing other natural settings in which the spanoid structure arises in the hope of motivating the questions raised in the context of LCCs and demonstrating their potential to contribute to research in other areas. We then turn to the investigation of functional rank of spanoids, which aims to convert them to actual LCCs. We conclude with describing some of the structural results about spanoids obtained here.

### 1.1 Locally Correctable Codes

The introduction of *locality* to coding theory has created a large body of research with wide-ranging applications and connections, from probabilistically checkable proofs, private information retrieval, program testing, fault-tolerant storage systems, and many others in computer science and mathematics. We will not survey these, and the reader may consult the surveys [27, 10]. Despite much progress, many basic questions regarding local testing, decoding and correcting of codes remain open. Here we focus on the efficiency of locally *correctable* codes, that we now define. Note that the related, locally *decodable* codes (LDCs), will not be discussed in this paper, as our framework is not relevant to them (LCCs can be converted to LDCs with a small loss in parameters).

► **Definition 1** (*q*-LCCs). A code  $C \subseteq \Sigma^n$  is called a *q*-query locally correctable code with error-tolerance  $\delta > 0$ , if for every  $i \in [n]$  there is a family (called a *q*-*matching*)  $M_i$ , of at least  $\delta n$  disjoint *q*-subsets of  $[n]$ , with the following *decodability* property. For every codeword  $c \in C$ , and for every  $i \in [n]$ , the value of  $c_i$  is *determined*<sup>5</sup> by the values of  $c$  in coordinates  $S$ , for every *q*-subset  $S$  in  $M_i$ .<sup>6</sup>

<sup>5</sup> Through some function that does not depend on the codeword  $c$ .

<sup>6</sup> Our definition is a ‘zero-error’ version of the standard definition. By ‘zero-error’ we mean that for any codeword  $c$ , the value of  $c_i$  can be determined correctly (without error) from the coordinates of  $c$  at any *q*-subset  $S$  in the matching  $M_i$ . A more general definition would say that  $c_i$  can be computed from  $c|_S$  with high probability, or even just slightly better than a random guess. Our definition is equivalent to the more general definition for linear codes, which comprise all of the interesting examples. We still allow ‘global’ error in the sense that a large (constant) fraction of the coordinates can be corrupted (this global error tolerance is captured by the parameter  $\delta$ ).

Intuitively, given a vector  $c' \in \Sigma^n$  which results from corrupting less than (say)  $\epsilon \delta n$  coordinates of a codeword  $c \in C$ , recovering  $c_i$  for any given  $i \in [n]$  is simple. Picking a random  $q$ -subset from  $M_i$  and decoding  $c_i$  according to it will succeed with probability at least  $1 - \epsilon$ , as only an  $\epsilon$ -fraction of these  $q$ -subsets can be corrupted.

We focus in this paper on the most well-studied and well-motivated regime where both the “query-complexity”  $q$  and the error-tolerance  $\delta$  are constants. It is not hard to see that there are no LCCs with  $q = 1$  (unless the dimension is constant) and so we will start with the first interesting case of  $q = 2$ . A canonical example of a 2-query LCC, which will serve us several times below, is the Hadamard code. Here  $\Sigma = \mathbb{F}_2$ . Let  $k$  be any integer and set  $n = 2^k - 1$ . Let  $A$  be the  $k \times n$  matrix whose columns are all non-zero  $k$ -bit vectors. The Hadamard code  $C_H \in \mathbb{F}_2^n$  is generated by  $A$ , namely  $H$  consists of all linear combinations of rows of  $A$ . Since every column of  $A$  can be written as a sum of (namely, spanned by) pairs of other columns in  $(n - 1)/2$  different ways, the matching  $M_i$  suggest themselves, and so is the *linear* correcting procedure: add the values in coordinates of the random pair  $S$  from  $M_i$  to determine the  $i$ th coordinate.

A central parameter of codes is their *rate*, capturing the redundancy between the *dimension*, namely the number of information bits encoded (here  $k$ ), and the length of the codeword (here  $n$ ). As in this paper this  $k$  will be a tiny function of  $n$ , we will focus on the *dimension* itself. Note that in the example above, as in every *linear* code, this dimension is also the *rank* of the generating matrix. In general codes, dimension may be fractional, and is defined as follows. All logarithms are in base 2 unless otherwise noted.

► **Definition 2** (Dimension and rate of a code). For a general, possibly non-linear code  $C \subseteq \Sigma^n$ , we define the dimension of  $C$  to be  $\dim(C) = \log |C| / \log |\Sigma|$ . Note that this coincides with the linear algebraic definition of dimension when  $C$  is a subspace. We refer to the ratio  $\dim(C)/n$  as the ‘rate’ of the code.

Note that while the Hadamard code ( $C_H$ ) has fantastic local correction (only 2 queries), its dimension is only  $k \sim \log n$ , which is pathetic from a coding theory perspective. However, no better 2-query LCC can exist, regardless of the alphabet.

► **Theorem 3** (2-LCCs). For all large enough  $n$  and over any alphabet:

- There exists a 2-query LCC of dimension  $\Omega(\log n)$  and constant  $\delta$  (Folklore: Hadamard code).
- Every 2-query LCC must have dimension at most  $O(\log n)$  (for any constant  $\delta$ ) [6].

While we know precisely the optimal dimension for 2 queries, for  $q \geq 3$  the gap between known upper and lower bounds is huge. The best lower bounds (constructions) are polylogarithmic: they come from Reed-Muller codes (using polynomials over finite fields), and yield dimension  $\Omega((\log n)^{q-1})$ .

The best LCC upper bounds are only slightly sub-linear, giving  $\dim(C) \leq \tilde{O}(n^{1-\frac{1}{q-1}})$  (up to logarithmic factors). This bound, which we will refer to as the Katz-Trevisan (KT) bound, is actually a slight refinement/improvement over the bound originally appearing in [19] (which gave  $n^{1-1/q}$ ). This improvement was implicit in several works (e.g. [9, 25]) and is explicitly stated in [18]. We should also note that, over constant-size alphabets, Kerenidis and De-Wolf proved an even stronger bound using quantum information theory [21]. This exponential gap between the upper and lower bounds, which we formally state below, has not been narrowed in over two decades.<sup>7</sup> Explaining this gap (in the hope of finding ways to close it) is one major motivation of this work.

<sup>7</sup> For LDCs better constructions than Reed-Muller codes are known, through the seminal works of [26, 12], but as mentioned we will not discuss them here. Still, the upper bounds for LDCs are the same as for

- **Theorem 4** ( $q$ -LCCs,  $q \geq 3$ ). For every fixed  $q \geq 3$  and all large enough  $n$ :
  - There exists a  $q$ -query LCC of dimension  $\Omega((\log n)^{q-1})$  (with constant  $\delta$  and alphabet of size  $q + 1$ ) (Reed-Muller codes, see e.g. the survey [27]).
  - Every  $q$ -query LCC must have dimension at most  $\tilde{O}(n^{1-\frac{1}{q-1}})$  (for any constant  $\delta$  and any alphabet) [18].

## 1.2 Spanoids

We shall now abstract the notion of inference used in LCCs. There, for a collection of pairs  $(S, i)$  with  $S \subseteq [n]$  and  $i \in [n]$ , the values of codewords in coordinate positions  $S$ , determine the value of some other coordinate  $i$ . We shall forget (for now) the underlying code altogether, and abstract this relation by the formal “inference” symbol  $S \rightarrow i$ , to be read “ $S$  spans  $i$ ”.

► **Definition 5** (Spanoid). A *spanoid*  $\mathcal{S}$  over  $[n]$  is a family of pairs  $(S, i)$  with  $S \subseteq [n]$  and  $i \in [n]$ . The pair  $(S, i)$  will sometimes be written as  $S \rightarrow i$  and read as  $S$  spans  $i$  in the spanoid  $\mathcal{S}$ .

One natural way to view a spanoid is as a logical inference system, with the pairs indicating all inference rules. The elements of  $[n]$  indicate some  $n$  formal statements, and an inference  $S \rightarrow i$  of the spanoid means that if we know the truth of the statements in  $S$ , we can infer the truth of the  $i$ th statement. With this intuition, we shall adopt the convention that the inferences  $i \rightarrow i$  are implicit in any spanoid, and that *monotonicity* holds: if  $S \rightarrow i$  then also  $S' \rightarrow i$  for every  $S' \supseteq S$ . These conventions will be formally stated below when we define general derivations, which sequentially combine these implicit rules and the stated rules (pairs) of the spanoid.

A key concept of spanoids is, naturally, the *span*. Given a subset  $T \subseteq [n]$  (which we can think of as “axioms”), we can explore everything they can span by a sequence of applications of the inference rules of the spanoid  $\mathcal{S}$ .

► **Definition 6** (Derivation, Span). A *derivation* in  $\mathcal{S}$  of  $i \in [n]$  from  $T \subseteq [n]$ , written  $T \models_{\mathcal{S}} i$ , is a sequence of sets  $T = T_0, T_1, \dots, T_r$  with  $i \in T_r$  such that for each  $j \in [r]$ ,  $T_j = T_{j-1} \cup i_j$  for some  $i_j \in [n]$  and there exists  $S \subset T_{j-1}$  such that  $(S, i_j) \in \mathcal{S}$  is one of the spanoid rules.

The *span* (or *closure*) of  $T$ , denoted  $\text{span}_{\mathcal{S}}(T)$ , is the set of all  $i$  for which  $T \models_{\mathcal{S}} i$ . We shall remove the subscript  $\mathcal{S}$  from these notations when no confusion about the underlying spanoid can arise, and write  $T \models i$  and  $\text{span}(T)$  for short.

Despite being highly abstract, we will see that spanoids can lead to a rich family of questions and definitions. The first, and perhaps one of the most central definitions is that of the *rank* of a spanoid. We shall see other notions of spanoid rank later on (and will discuss the relation between them).

► **Definition 7** (Rank). The *rank* of a spanoid  $\mathcal{S}$ , denoted  $\text{rank}(\mathcal{S})$ , is the size of the *smallest* subset  $T \subseteq [n]$  such that  $\text{span}(T) = [n]$ . Note that by the definition of span we always have  $\text{rank}(\mathcal{S}) \leq n$ .

We note that the “rank” of a logical inference system does appear (under different names) in *proof complexity*. It is the starting point for *expansion-based* lower bounds on a variety of proof systems, as introduced for Resolution proofs in [5], and used for many others e.g. in [1] and [2]). We shall return to this connection presently.

---

LCCs, and obtained by the same KT-type argument, so the results in this paper may serve to better understand the (smaller, but still quite large) gap between upper and lower bounds in LDCs as well.



We can now define the spanoid analog of  $q$ -LCCs as spanoids which only specify the correction structure (the matchings  $M_i$ ) without requiring any codewords or alphabet.

► **Definition 8** ( $q$ -LCS, Locally correctable spanoid). A spanoid  $\mathcal{S}$  over  $[n]$  is a  $q$ -LCS with error-tolerance  $\delta$  if for every  $i \in [n]$  there exists a family  $M_i$  of at least  $\delta n$  disjoint  $q$ -subsets of  $[n]$  such that for each  $S \in M_i$  we have  $(S, i) \in \mathcal{S}$ . Namely, each  $i \in [n]$  is spanned (in  $\mathcal{S}$ ) by at least  $\delta n$  disjoint subsets of  $q$ -elements.

One can now ask about the highest possible rank of a  $q$ -LCS. It is not hard to see that the existence of a  $q$ -LCC (over any alphabet)  $C \subset \Sigma^n$  with dimension  $\dim(C) = d$  automatically implies that there exists a  $q$ -LCS (namely, the one given by the same matchings used in  $C$ ) with rank at least  $\lceil d \rceil$ . Indeed, otherwise there would be  $r < d$  coordinates in  $[n]$  that determine any codeword  $c \in C$  and this would limit the number of codewords to  $\Sigma^r$ .

One of our main observations is that, remarkably, in locally correctable spanoids there is *no* gap between the upper and lower bounds: we know the precise answer up to logarithmic factors, and it matches the *upper bounds* for LCCs! Observe the analogies to the theorems in the previous subsection, for  $q = 2$  and  $q \geq 3$ .

► **Theorem 9** (2-LCSs). *For all large enough  $n$ :*

- *There exists a 2-LCS over  $[n]$  with error-tolerance  $\delta$  of rank  $\Omega(\frac{1}{8} \log(\delta n))$ .*
- *Every 2-LCS over  $[n]$  with error-tolerance  $\delta$  must have rank at most  $O(\frac{1}{8} \log(n))$ .*<sup>8</sup>

Here, of course, the inference structure of the Hadamard code proves the first item. To get the required dependence on  $\delta$ , one can take  $\frac{1}{8}$  disjoint copies of such spanoids. The second item requires a new proof we discuss below, which generalizes (and implies) the one in Theorem 3. It is quite surprising that, even in this abstract setting, with no need for codewords or alphabet, one cannot do better than the Hadamard code!

We now state our results for  $q \geq 3$ .

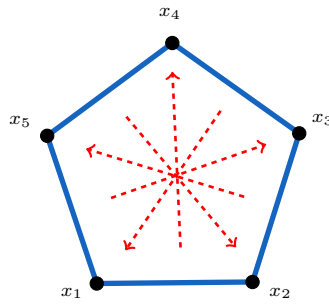
► **Theorem 10** ( $q$ -LCSs with  $q \geq 3$ ). *For every fixed  $q \geq 3$  and all large enough  $n$ :*

- *There exist a  $q$ -LCS of rank  $\tilde{\Omega}(n^{1-\frac{1}{q-1}})$  (with constant  $\delta$ ).*
- *Every  $q$ -LCS over  $[n]$  has rank at most  $\tilde{O}(n^{1-\frac{1}{q-1}})$  (for any constant  $\delta$ ).*

Both parts of this theorem demand discussion. The possibly surprising (and tight) lower bound follows from a simple probabilistic argument (indeed, one which is repeatedly used to prove expansion in the proof complexity references cited above), where the matchings  $M_i$  are simply chosen uniformly at random. It seems to reveal how significant a relaxation spanoids are of LCCs (where probabilistic arguments fail completely). However, the best known LCC upper bound (Theorem 4) does not rule out the possibility that, at least for large alphabets, the two (LCC's dimension and LCS's rank) have the same behavior! From a more pessimistic (and perhaps more realistic) perspective, our lower bound shows the limitations of any (upper bound) proof technique which, in effect, applies also for spanoids. These are proofs in which the LCC structure is used to show that a small subset spans all the others. We note that there are several LCC upper bounds which 'beat' the  $n^{1-\frac{1}{q-1}}$  bound for certain very special cases by using additional structure not present in the corresponding abstract spanoid. One example is the bound of [21], which uses arguments from quantum information theory to roughly *halve* the number of queries, over binary (or small) alphabets. Another example is

<sup>8</sup> The results of [6] can be interpreted as an upper bound of  $O(\text{poly}(1/\delta) \log(n))$  on the rank of 2-LCS with error-tolerance  $\delta$ .





■ **Figure 1** The pentagon spanoid  $\Pi_5$  where each coordinate is spanned by the coordinates of the opposite edge.

the paper [11], which gives an improved upper bound on the dimension for linear 3-LCCs defined over the real numbers, using specific properties of the Reals such as distance and volume arguments.

Our proof of the upper bound, is again more general than for LCCs, and interesting in its own right. We use a simple technique which performs random restrictions and contractions of graphs and hypergraphs (and originates in [11]). It will be described in Section 3, after we have formulated an equivalent, set-theoretic formulation of spanoids in Section 2.2.

### 1.2.1 Functional rank: bridging the gap between LCCs and LCSs

We conclude this section of the introduction with an attempt to understand (and possibly bridge) the gap between LCCs and their spanoid abstraction. The idea is to start with an LCS of high rank (which we know is possible), and convert it to an LCC without losing too much in the parameters. More generally, for a given spanoid  $\mathcal{S}$ , we would like to investigate the code  $C$  with largest dimension (over any alphabet  $\Sigma$ ) which would be consistent with the inferences of  $\mathcal{S}$ . This is captured in the notion of *functional rank* which we now define.

► **Definition 11** (Functional rank). Let  $\mathcal{S}$  be a spanoid over  $[n]$ . A code  $C \subset \Sigma^n$  is *consistent* with  $\mathcal{S}$  if for every inference  $(S, i)$  in  $\mathcal{S}$ , and for every codeword  $c \in C$ , its values of coordinates  $S$  determine its value in coordinate  $i$  (by some fixed function,  $f_{S,i}$  not depending on  $c$ ).<sup>9</sup>

Define the *functional rank* of  $\mathcal{S}$ , denoted  $\text{f-rank}(\mathcal{S})$ , to be equal to the supremum of the dimension  $\dim(C)$ , over all possible finite alphabets  $\Sigma$  and codes  $C \subset \Sigma^n$  which are consistent with  $\mathcal{S}$ .

Of course, the strategy of constructing LCCs in two stages as above can only work if we can bound the gap between  $\text{rank}(\mathcal{S})$  and  $\text{f-rank}(\mathcal{S})$ . This question, of bounding this gap or proving it can be large, is perhaps the most interesting one we raise (and leave mostly open for now). For now, we are able to show an example in which the two are different. The example providing a gap is depicted in Figure 1, arranging the coordinates as the vertices of a pentagon, the pair of vertices of each edge span the vertex opposite to it. That is,  $\{x_1, x_2\} \rightarrow x_4, \{x_2, x_3\} \rightarrow x_5$  etc.

► **Theorem 12** (Constant gap between rank and functional rank). *The pentagon spanoid  $\Pi_5$  depicted in Figure 1 has  $\text{rank}(\Pi_5) = 3$  but  $\text{f-rank}(\Pi_5) = 2.5$ .*

<sup>9</sup> One can think of a code consistent with  $\mathcal{S}$  also as a ‘representation’ of  $\mathcal{S}$  in the spirit of matroid theory.

Seeing that  $\text{rank}(\Pi_5) = 3$  is easy by inspection. The lower bound of 2.5 on functional rank comes from a set-theoretic construction of consistent codes. This comes from a simple linear programming (LP) relaxation we develop for  $\text{rank}(\mathcal{S})$  called  $\text{LP}^{\text{cover}}(\mathcal{S})$ , but surprisingly this LP captures the best set-theoretic construction of consistent codes. But even in this small example, the upper bound on functional rank is nontrivial to determine, as we allow all possible alphabets and consistent codes. Not surprisingly, Shannon entropy is the key to proving such a bound. We develop a linear programming relaxation, based on entropy whose optimum  $\text{LP}^{\text{entropy}}(\mathcal{S})$  upper bounds  $\text{f-rank}(\mathcal{S})$ . In this example, it proves 2.5 to be the optimum. For the definitions of  $\text{LP}^{\text{entropy}}(\mathcal{S})$ ,  $\text{LP}^{\text{cover}}(\mathcal{S})$  and the proof of Theorem 12 see the full version. One natural way of amplifying gaps as in the example above, which may also be useful in creating codes of high functional rank, is the idea of *tensoring*. We develop different notions of tensoring *spanoids* inspired by tensoring of codes. In particular, we define a product of spanoids called the *semi-direct product* under which  $\text{rank}$  is multiplicative and  $\text{f-rank}$  is sub-multiplicative. By repeatedly applying this product to  $\Pi_5$ , we get a spanoid with polynomial gap between  $\text{f-rank}$  and  $\text{rank}$ . See the full version for details.

► **Theorem 13** (Polynomial gap between rank and function rank). *There exists a spanoid  $\mathcal{S}$  on  $n$  elements with  $\text{rank}(\mathcal{S}) \geq n^c \text{f-rank}(\mathcal{S})$  where  $c = \log_5 3 - \log_5 2.5 \geq 0.113$ .*

Summarizing, we have the following obvious inequalities between the measures we described so far for every spanoid  $\mathcal{S}$ . We feel that understanding the exact relationships better is worthy of further study

$$\text{LP}^{\text{cover}}(\mathcal{S}) \leq \text{f-rank}(\mathcal{S}) \leq \text{LP}^{\text{entropy}}(\mathcal{S}) \leq \text{rank}(\mathcal{S}). \quad (1)$$

### 1.3 Other motivations and incarnations of spanoids

We return to discuss other structures, combinatorial, geometric and algebraic, in which the same notions of span and inference naturally occur, leading to a set-theoretic one that elegantly captures spanoids precisely. These raise further issues, some of which we study in this paper and some are left for future work. These serves to illustrate the breadth of the spanoid framework.

#### 1.3.1 Bootstrap percolation and gossip processes

The following general set-up occurs in statistical physics, network theory and probability theory. Fix an undirected graph  $G([n], E)$ . In a gossip or infection process, or equivalently bootstrap percolation, we are given a set of “rules” specifying, for every vertex  $v \in [n]$ , a family of subsets of its neighbors. The intended meaning of such a rule is that if *every* member of one such subset is “infected” at a certain time step, then the vertex  $v$  becomes infected in the next time step. Given a set of initial infected vertices, this defines a process in which infection spreads, and eventually stabilizes. A well studied special case is the (uniform) *r-bond percolation* [7], where the family for each vertex is all  $r$ -subsets of its neighbors. Many variants exist, e.g. one can have a similar process on the edges, rather than vertices of the graph. An important parameter of such a process is the following: what is the size of the smallest set of vertices which, if infected, will eventually infect all other vertices<sup>10</sup>.

<sup>10</sup>This turns out to be crucial for understanding, at least for certain structured graphs like lattices studied by physicists, the threshold probability for percolation when initial infections are random.

A moment’s thought will convince the reader that this structure is precisely a spanoid (where inferring sets are restricted by the graph structure). The infection process is precisely the inference process defining span in spanoids. Furthermore, the smallest size of an infecting set is *precisely* the rank of that spanoid! Much work has been invested to determine that rank even in very special cases, e.g. for the  $r$ -bond percolation above, in e.g. Boolean hypercubes, where it is known precisely. Interestingly, the paper [16] uses the so-called “polynomial method” to reprove that bound, which fits even deeper with our framework. In our language, their method determines the *functional rank* of this spanoid, and one direction is through constructing an explicit code that is consistent with the spanoid! The reader is encouraged to work out the details.

### 1.3.2 Independence systems and Matroids

An *independence system* over  $[n]$  is a family  $\mathcal{F}$  of subsets of  $[n]$  which is downwards-closed (if a set is in  $\mathcal{F}$ , so are all its subsets). The members of  $\mathcal{F}$  are called *independent*. While much of what we say below generalizes to all independence systems, we specify them for the important special systems called *matroids*.

A *matroid* is an independence system in which the independent sets satisfy the so-called “exchange axiom” (which we will not define here). Matroids abstract *linear* independence in subsets of a vector space over a field<sup>11</sup>, and capture algorithmic problems in which optimization is possible through the greedy algorithm. Matroids thus come with natural notions of span and rank, extending the ones in the linear algebraic setting. The rank of a set is the size of the largest independent set it contains. The span of a set is the maximal superset of it of the same rank. A matroid can thus be naturally viewed as a spanoid, with the inference rules  $F \rightarrow i$  for every independent  $F \in \mathcal{F}$  and every  $i$  for which  $F \cup \{i\}$  is *not* independent (such minimal dependent sets as  $F \cup \{i\}$  are called *cycles*). It is easy to verify that the notions of span and rank of the matroid and the spanoid it defines coincide. This also raises the natural question of bounding the gap between **f-rank** and **rank** for the special case of spanoids arising from matroids.

Note that a spanoid resulting from a matroid this way is *symmetric*: by the exchange property of matroids, if  $E \subset [n]$  is a cycle of  $\mathcal{F}$ , then for *every*  $i \in E$  it contains the inference  $E \setminus \{i\} \rightarrow i$ . Symmetric spanoids are interesting, and we note that the pentagon example witnessing the gap between rank and functional rank is *not* symmetric, and we do not know such a gap for symmetric spanoids. We also don’t know if symmetric spanoids can achieve the lower bound in Theorem 10.

### 1.3.3 Point-line incidences

Sylvester-Gallai theorem is a celebrated result in combinatorial geometry conjectured by Sylvester and proved independently by Melchior and Gallai. It states that for any set of  $n$  points in Euclidean space  $\mathbb{R}^d$ , if the line through any two points passes through a third point, then they must all be collinear (namely, they span a 1-dimensional affine space). Over the complex numbers, one can prove a similar theorem but with the conclusion that the points span a 2-dimensional affine space (and there are in fact two dimensional examples known) [20]. Over finite fields the conclusion is even weaker, saying that the span has dimension at most  $O(\log(n))$  and this is tight as the example of all points in  $\mathbb{F}_p^k$  with  $n = p^k$  shows. It

<sup>11</sup> Matroids are in fact more general than linear independent sets of vectors over a field, for example the Vámos matroid on eight elements is not representable over any field.

is not a coincidence that this example reminds one of the Hadamard code described before as an example of a 2-query LCC. It is in fact true that there is a tight connection between configurations of points with many collinear triples and linear 2-query LCCs. This was first noticed in [4, 3] and was used to prove that 2-LCCs do not exist over the characteristic zero fields (for  $q \geq 3$  these questions are wide open with even larger gaps than in the finite field case). LCCs with more than 2 queries naturally correspond to point configurations with many  $(q - 2)$ -dimensional affine spaces containing at least  $q$  points.

Given the connections between Sylvester-Gallai type incidence structures and LCCs, and the insights offered by spanoids for studying LCCs, it is natural that the study of the spanoid structures can help us get new insights on incidence geometry problems. A *dual* way to view these incidences, which we shall presently generalize, is to consider each point  $p_i : i \in [n]$  as representing a hyperplane  $F_i$  through the origin (in the appropriate vector space) vanishing on the linear function defined by  $p_i$ . A point  $p_i$  is spanned by a collection of points  $\{p_j : j \in S\}$  iff  $F_i \supset \bigcap_{j \in S} F_j$ . Therefore the spanning structure of the points  $p_1, p_2, \dots, p_n$  is captured by the spanoid where we would add the inference  $S \rightarrow i$  iff  $F_i$  contains the common intersection of all  $F_j : j \in S$ .

### 1.3.4 Systems of polynomial equations

Given the above example, there is no reason to stop at the linear setting. Instead of lines we can consider  $n$  (multivariate) polynomials  $f_i$  over a field, and again consider  $F_i$  to be the zero set of  $f_i$ . The spanoid above, having an inference  $S \rightarrow i$  whenever the set  $F_i$  contains the common intersection of all  $F_j : j \in S$ , is capturing another natural algebraic notion. Namely, it says that the polynomial  $f_i$  vanishes on all the common roots of the polynomials  $\{f_j : j \in S\}$ . By the celebrated Hilbert's Nullstellensatz theorem, over algebraically closed fields, this implies that  $f_i$  belongs to the radical of the ideal generated by the  $f_j$ 's. Here the rank function is far from being that of a matroid; the complex spanoid which arises (and in general is far from understood) plays a role in arithmetic complexity (a beautiful example is the recent [23] dealing with degree-2 polynomials).

### 1.3.5 Intersecting set systems

Let us remove all restrictions from the origin or nature of the  $n$  sets  $F_i$  discussed in the previous discussion. Assume we are given any such family  $\mathcal{F}$  of sets (from an arbitrary universe, say  $U$ ). As above, a natural spanoid  $\mathcal{S}_{\mathcal{F}}$  will have the inference  $S \rightarrow i$  whenever the set  $F_i$  contains the common intersection of all  $F_j : j \in S$ . Such situations (and hence, spanoids) arise in many questions of extremal set theory, for example the study of (weak) sunflowers, or families with certain forbidden intersection (or union) patterns, e.g. [14, 13, 15].

What is interesting in this more general framework, where the initial family of sets  $\mathcal{F}$  is arbitrary, is that it becomes *equivalent* to spanoids! In other words, every spanoid  $\mathcal{S}$  arises as  $\mathcal{S}_{\mathcal{F}}$  of *some* family of sets  $\mathcal{F}$ . This possibly surprising fact is not much more than an observation, but it turns out to be an extremely useful formulation for proving some of the results in this paper. Let us state it formally (it will be proved in Section 2.2).

► **Theorem 14** (Spanoids and intersecting sets). *Let  $\mathcal{S}$  be any spanoid on  $[n]$ . Then, there exists a universe  $U$  and a family  $\mathcal{F}$  of  $n$  sets of  $U$ ,  $\mathcal{F} = \{F_1, F_2, \dots, F_n\}$ , such that  $\mathcal{S} = \mathcal{S}_{\mathcal{F}}$ .*

It is convenient to assume that the sets in  $\mathcal{F}$  have no element in common to all<sup>12</sup>.

The notions of rank and span are extremely simple in this set-theoretic setting, and do not require the sequential “derivation” and the implicit ordering which we require to define these in spanoids. For a family  $\mathcal{F}$  of  $n$  sets and a subset  $S \subset [n]$ , let us denote by  $\cap S$  the subset of  $U$  which is the intersection of all  $\{F_j : j \in S\}$ . Then the rank of  $S$  is the size of the smallest subset  $S' \subseteq S$  for which  $\cap S' = \cap S$ . Similarly, the span of  $S$  is the largest superset  $S'' \supseteq S$  for which  $\cap S'' = \cap S$ .

These static definitions of rank and span make many things transparent. For example, the expected fact that testing if the rank of a spanoid (namely the rank of the set  $[n]$ ) is at most some given integer  $k$  is *NP-complete* (Claim 22). Complementing the sets  $F_i$  in  $\mathcal{F}$ , and replacing intersection with union, this is precisely the Set Cover problem. This connection also underlies the cover-based linear program discussed earlier, as well as proofs of the main quantitative results Theorem 9 and Theorem 10.

### 1.3.6 Union-closed families

Spanoids over  $[n]$  are equivalent to union-closed families of subsets of  $[n]$  i.e. a family of subsets of  $[n]$  such that the union of any two members is again in the family. A closed set of a spanoid  $\mathcal{S}$  is a subset  $A \subset [n]$  such that  $\text{span}(A) = A$ . The family of all closed sets of a spanoid  $\mathcal{S}$  is denoted by  $\mathcal{C}_{\mathcal{S}}$  which is an intersection-closed family. Thus the family of all open sets which are complements of closed sets is a union-closed family and is denoted by  $\mathcal{O}_{\mathcal{S}}$ . One can construct all the derivations of the spanoid given its family of open or closed sets. Conversely, given any union-closed family of subsets of  $[n]$ , one can define a spanoid whose open sets are precisely the given family. Thus spanoids on  $[n]$  are equivalent to union-closed families of subsets of  $[n]$ . The rank of a spanoid has a very simple interpretation in terms of its open sets,  $\text{rank}(\mathcal{S})$  is equal to the size of the smallest hitting set for its family of open sets  $\mathcal{O}_{\mathcal{S}}$ . Moreover,  $\text{rank}(\mathcal{S})$  is at most  $\log |\mathcal{O}_{\mathcal{S}}|$ . These connections are discussed in Section 2.1.

Union-closed families are interesting combinatorial objects with a rich structure. The widely open Frankl’s union-closed conjecture states that in every union-closed family of  $N$  sets, there exists an element which is contained in at least  $N/2$  sets. Though this was proved for various special classes (see survey [8]), the best general bound is  $\Omega(N/\log N)$  due to [22, 24]. When seen in the framework of spanoids, this follows immediately from Claim 19 which says that there is a  $\log N$  sized hitting set for every union-closed family of  $N$  sets. Thus there is an element which should hit at least  $N/\log_2(N)$  sets. We hope that viewing union-closed families as spanoids could be of use in understanding them.

## 1.4 Organization

In Section 2, we will present two alternative ways to represent spanoids that will be very useful. In Section 3, we show upper bounds on the rank of  $q$ -LCSs for  $q \geq 2$  thus proving the upper bounds in Theorems 27 and 29. In Section 4, we construct  $q$ -LCSs thus proving the lower bound in Theorem 29.

---

<sup>12</sup>Indeed, otherwise we can remove the common intersection (if non-empty) of *all* members of  $\mathcal{F}$  from each of them, as it does not change the underlying spanoid.

## 2 Preliminaries on spanoids

We will now describe two equivalent ways to define spanoids which will turn out to be very useful.

### 2.1 Spanoids as union-closed or intersection-closed families

In this subsection, we will define an equivalent and more canonical way of describing spanoids in terms of intersection closed or union closed families. We have defined spanoids in Definition 5 by specifying the initial set of derivation rules such as  $A \rightarrow i$ . But two different initial set of rules can lead to the same set of derivations and we should consider two spanoids to be equivalent if they lead to the same set of derivations. We will present an alternative way to describe spanoids which makes them equivalent to union-closed families of sets or alternatively intersection-closed families of sets. Moreover this new representation is a more canonical way to represent spanoids since it will be based only on the set of derivations. For this, the main new notions we need to define are that of a ‘closed set’ and an ‘open set’.

► **Definition 15.** (Closed and open sets) Let  $\mathcal{S}$  be a spanoid on  $[n]$ . A *closed set*<sup>13</sup> is a subset  $B \subset [n]$  for which  $\text{span}(B) = B$ . A subset  $B \subset [n]$  is called an *open set* if its complement is a closed set. The family of all closed sets of  $\mathcal{S}$  is denoted by  $\mathcal{C}_{\mathcal{S}}$  and the family of all open sets of  $\mathcal{S}$  by  $\mathcal{O}_{\mathcal{S}}$  (when it is clear from the context, we will drop the subscript  $\mathcal{S}$ ).

► **Claim 16.** In any spanoid  $\mathcal{S}$  on  $[n]$ ,

1. the intersection of any number of closed sets is a closed set i.e.  $\mathcal{C}_{\mathcal{S}}$  is an intersection-closed family,
2. the union of any number of open sets is an open set i.e.  $\mathcal{O}_{\mathcal{S}}$  is a union-closed family and
3. for any set  $A \subset [n]$ ,  $\text{span}(A)$  is equal to the intersection of all closed sets containing  $A$  i.e.  $\text{span}(A) = \bigcap_{B \supset A, B \in \mathcal{C}_{\mathcal{S}}} B$ .

**Proof.**

- (1) Let  $F = F_1 \cap F_2$  be the intersection of two closed sets. Suppose in contradiction that  $F$  spans some element  $x \in [n] \setminus F$  then, by monotonicity, both  $F_1$  and  $F_2$  have to span  $x$ . Hence,  $x \in \text{span}(F_1) \cap \text{span}(F_2) = F_1 \cap F_2 = F$  in contradiction.
- (2) This just follows from (1) by taking complements.
- (3) Let  $F(A)$  be the intersection of all closed sets containing  $A$ . Since  $\text{span}(A)$  is a closed set we clearly have  $F(A) \subset \text{span}(A)$ . To see the other direction, suppose  $x \in \text{span}(A)$  and let  $F$  be any closed set containing  $A$ . Then, by monotonicity,  $F$  must also span  $x$  and so we must have  $x \in F$ . ◀

► **Claim 17.** A spanoid is uniquely determined by the set of all its closed (open) sets which is an intersection-closed (union-closed) family of subsets of  $[n]$ . Conversely, every intersection-closed (union-closed) family of subsets of  $[n]$  defines a spanoid whose closed (open) sets are the given family.

**Proof.** Given a spanoid  $\mathcal{S}$  on  $[n]$ , by Claim 16, we can define  $\text{span}(A)$  in  $\mathcal{S}$  using just the closed sets as:

$$\text{span}(A) = \bigcap_{B \supset A, B \in \mathcal{C}_{\mathcal{S}}} B.$$

<sup>13</sup>Closed sets are analogous to ‘flats’ or ‘subspaces’ in matroids.

And  $A \models i$  in  $\mathcal{S}$  iff  $i \in \text{span}(A)$ . Thus given the set of all closed sets, we can reconstruct all the derivations of the spanoid.

For the converse, suppose we are given an intersection-closed family of subsets of  $[n]$ , say  $\mathcal{C}$ . We can define  $\text{span}_{\mathcal{C}}(A) = \bigcap_{B \supset A, B \in \mathcal{C}} B$  and define a spanoid  $\mathcal{S}_{\mathcal{C}}$  where  $A \models i$  iff  $i \in \text{span}_{\mathcal{C}}(A)$ . It is easy to see that the closed sets of this spanoid  $\mathcal{S}_{\mathcal{C}}$  is exactly  $\mathcal{C}$ . ◀

Thus an equivalent way to define a spanoid is to define all its closed (open) sets which is some intersection (union) closed family. The following claim shows that the rank of a spanoid has a very natural interpretation in terms of the open sets.

► **Claim 18.** *The rank of a spanoid  $\mathcal{S}$  is the size of the smallest hitting set for the collection  $\mathcal{O}_{\mathcal{S}}$  i.e. a set which intersects every open set in  $\mathcal{O}_{\mathcal{S}}$  non-trivially.*

**Proof.** Observe that a subset  $A \subset [n]$  spans  $[n]$  iff it is a hitting set for all the open sets in  $\mathcal{O}_{\mathcal{S}}$ . This is because if  $A$  doesn't hit some open set  $B$ , then  $A$  lies in the complement of  $B$  i.e.  $A \subset \bar{B}$ . Since  $\bar{B}$  is closed,  $\text{span}(A) \subset \bar{B} \neq [n]$ . Therefore  $\text{rank}(\mathcal{S})$  is the size of the smallest hitting set for  $\mathcal{O}_{\mathcal{S}}$ . ◀

This interpretation of the rank is used to give a linear programming relaxation  $\text{LP}^{\text{cover}}$  which lower bounds the rank. We can also upper bound the rank of a spanoid in terms of the number of closed or open sets as the following claim shows.

► **Claim 19.** *Let  $\mathcal{S}$  be a spanoid, then  $\text{rank}(\mathcal{S}) \leq \log_2(|\mathcal{C}_{\mathcal{S}}|) = \log_2(|\mathcal{O}_{\mathcal{S}}|)$ .*

**Proof.** Let  $r = \text{rank}(\mathcal{S})$  and  $R \subset [n]$  be a set of size  $|R| = r$  spanning  $[n]$ . Since the rank of  $\mathcal{S}$  is  $r$  we know that  $R$  is independent (not spanned by any proper subset). For each of the  $2^r$  subsets  $S \in 2^R$  we consider the closed set  $F_S = \text{span}(S)$ . We claim that all of these are distinct. Suppose in contradiction that there were two distinct sets  $S \neq T \in 2^R$  with  $\text{span}(S) = \text{span}(T)$ . W.l.o.g suppose there is an element  $x \in T \setminus S$ . Then  $x \in \text{span}(S)$  and so we get that  $R \setminus \{x\}$  spans  $R$  (by monotonicity) and so spans the entire spanoid in contradiction. Thus  $|\mathcal{C}_{\mathcal{S}}| \geq 2^r$ . ◀

## 2.2 Spanoids as set systems

In this subsection, we will show yet another way of representing spanoids by families of sets. This representation (which is equivalent to spanoids) will be easier to work with and, in fact, we will later work almost exclusively with it instead of with the definition given in the introduction. Recall the notation introduced at the end of the introduction that, for sets  $S_1, \dots, S_n$  and for a subset  $A \subset [n]$  we let  $\cap A = \bigcap_{i \in A} S_i$ .

► **Definition 20** (Intersection Dimension of a set system). The *intersection-dimension* of a family of sets  $S_1, \dots, S_n$ , denoted  $\text{idim}(S_1, \dots, S_n)$  is the smallest integer  $d$  such that there exist a set  $A \subset [n]$  of size  $d$  such that  $\cap A = \cap [n]$ .

► **Lemma 21** (Set-Representation of spanoids). *Let  $\mathcal{S}$  be a spanoid on  $[n]$  with  $\text{rank}(\mathcal{S}) = r$ . Then there exists a family of sets  $S_1, \dots, S_n$  such that  $A \models i$  in  $\mathcal{S}$  iff  $\cap A \subset S_i$ . In this case we say that the set family  $(S_1, S_2, \dots, S_n)$  is a set-representation of  $\mathcal{S}$  and this implies in particular that  $\text{idim}(S_1, \dots, S_n) = \text{rank}(\mathcal{S})$ .*

**Proof.** For  $i \in [n]$  we define  $S_i \subset \mathcal{C}_{\mathcal{S}}$  to be the subfamily of closed sets of  $\mathcal{S}$  containing the element  $i \in [n]$ . For the first direction of the proof suppose that  $A$  spans  $x$  in the spanoid  $\mathcal{S}$ . Then, by Claim 16,  $x$  belongs to any closed set containing  $A$  and so  $\bigcap_{i \in A} S_i \subset S_x$ . For the other direction, suppose  $\bigcap_{i \in A} S_i \subset S_x$  or that any closed set containing  $A$  must also contain  $x$ . Hence,  $x$  is in the intersection of all closed sets containing  $A$  and, by Claim 16 we have that  $x \in \text{span}(A)$ . ◀



An alternative way to represent spanoids is by unions.  $(T_1, T_2, \dots, T_n)$  is called a *union set-representation* of the spanoid  $\mathcal{S}$  when,  $A \models i$  in  $\mathcal{S}$  iff  $T_i \subset \cup_{j \in A} T_j$ . Note that if  $(S_1, S_2, \dots, S_n)$  is an (intersection) set-representation for  $\mathcal{S}$  as in Lemma 21, then by taking complements,  $(\bar{S}_1, \bar{S}_2, \dots, \bar{S}_n)$  is a union set-representation for  $\mathcal{S}$  and vice versa. Thus these two notions of representing a spanoid by sets is equivalent.

► **Claim 22.** *Given a spanoid  $\mathcal{S}$  and some positive integer  $k$ , deciding if the rank of the spanoid is at most  $k$  is NP-complete.*

**Proof.** Given the description of a spanoid and a subset of its elements, we can check in polynomial time whether the subset has size at most  $k$  and spans all the elements. So the problem is in NP. To prove that it is NP-complete, we reduce Set Cover problem to this.

Given a collection of sets  $S_1, S_2, \dots, S_n \subset U$  such that  $\cup_i S_i = U$  and some positive integer  $k$ , the Set Cover problem asks if there are at most  $k$  sets in the collection whose union is  $U$ . To reduce it to the spanoid rank problem, we can create a spanoid over  $[n]$  elements where the inference rules are given by  $A \models i$  iff  $\cup_{j \in A} S_j \supset S_i$ . The rank of this spanoid is at most  $k$  iff there exists  $k$  sets in the collection which cover all of  $U$ . ◀

### 3 Upper bounds on the rank of $q$ -LCSs

In this section we prove the upper bounds on the rank of  $q$ -LCSs stated in Theorems 9 and 10. The proofs will rely on the set representation described in Section 2.2 and on random restriction and contraction arguments given below.

#### 3.1 Graph theoretic lemmas

In this subsection, we will prove a key technical lemma about a random graph process that will be useful for proving upper bounds on the rank of  $q$ -LCSs. We denote by  $\mathcal{D}(n)$  the set of simple directed graphs on  $n$  vertices. We always assume w.l.o.g that the set of vertices are the integers between 1 and  $n$ .

► **Definition 23** ( $(\alpha, \beta)$ -spread distribution). Let  $\mu$  be a distribution on  $\mathcal{D}(n)$ . We say that  $\mu$  is  $(\alpha, \beta)$ -spread if the following conditions are true for a graph  $G$  sampled from  $\mu$ :

1. Each vertex  $i \in [n]$  has an incoming edge with probability at least  $\alpha$  i.e.

$$\forall i \Pr_{G \sim \mu} [\exists j : (j, i) \in E(G)] \geq \alpha.$$

2. For every  $i, j \in [n]$ , the probability that  $(j, i)$  is an edge is at most  $\beta/n$  i.e.

$$\forall i, j \Pr_{G \sim \mu} [(j, i) \in E(G)] \leq \frac{\beta}{n}.$$

For example, one can generate an  $(k/n, 1)$ -spread distribution  $\mu$  on  $\mathcal{D}(n)$  in the following way: Fix arbitrary sets  $S_1, \dots, S_n \subset [n]$  of size  $k$  each. To sample a graph  $G$  from  $\mu$ , pick a uniformly random element  $j \in [n]$  and let  $G$  be the directed graph containing the edges  $(j, i)$  for each  $i$  such that  $j \in S_i$ . This satisfies the definition since for any fixed  $i \in [n]$ ,  $i$  has an incoming edge if  $j \in S_i$  which happens with probability  $|S_i|/n = k/n$ . And for any fixed  $i', j' \in [n]$ , the probability that  $(j', i')$  is an edge is at most  $1/n$  since this happens only when  $j' = j$  and  $j$  is chosen uniformly at random from  $[n]$ . Note that the sampled edges overall are highly correlated (they all have  $j$  as an endpoint).

We will need following simple observation about  $(\alpha, \beta)$ -spread distributions.

► **Lemma 24.** *Let  $\mu$  be an  $(\alpha, \beta)$ -spread distribution on  $\mathcal{D}(n)$ . For every vertex  $i$  and every subset  $S \subset [n]$  of size at most  $\frac{\alpha n}{2\beta}$ ,*

$$\Pr_{G \sim \mu}[\exists j \notin S : (j, i) \in E(G)] \geq \frac{\alpha}{2}.$$

**Proof.** This follows from union bound and properties of  $(\alpha, \beta)$ -spread distributions.

$$\begin{aligned} \alpha &\leq \Pr_{G \sim \mu}[\exists j : (j, i) \in E(G)] \\ &\leq \Pr[\exists j \in S : (j, i) \in E(G)] + \Pr[\exists j \notin S : (j, i) \in E(G)] \\ &\leq \sum_{j \in S} \Pr[(j, i) \in E(G)] + \Pr[\exists j \notin S : (j, i) \in E(G)] \\ &\leq \frac{\alpha n}{2\beta} \cdot \frac{\beta}{n} + \Pr[\exists j \notin S : (j, i) \in E(G)] \\ &= \frac{\alpha}{2} + \Pr[\exists j \notin S : (j, i) \in E(G)] \quad \blacktriangleleft \end{aligned}$$

Given a distribution  $\mu$  on graphs we would like to study the random process in which we, at each iteration, sample from  $\mu$  and ‘add’ the edges we got to the graph obtained so far. For two graphs  $G$  and  $H$  on the same set of vertices, we denote by  $G \cup H$  their set theoretic union (as a union of edges).

► **Definition 25** (Graph process associated with  $\mu$ ). Let  $\mu$  be a distribution on  $\mathcal{D}(n)$ . We define a sequence of random variables  $G_t^\mu$ ,  $t = 0, 1, 2, \dots$  as follows.  $G_0^\mu$  is the empty graph on  $[n]$  vertices. At each step  $t \geq 1$  we sample a graph  $G$  according to  $\mu$  (independently from all previous samples) and set  $G_t = G_{t-1} \cup G$ .

For a graph  $G \in \mathcal{D}(n)$  and a vertex  $i \in [n]$  we denote by  $\text{Rea}(i)$  the set of vertices that are reachable from  $i$  (via walking on directed edges). By convention, a vertex is always reachable from itself. Similarly, for a set of vertices  $S \subset [n]$  we denote by  $\text{Rea}(S) = \cup_{i \in S} \text{Rea}(i)$  the set of vertices reachable from some vertex in  $S$ . We denote the set of strongly connected components of  $G$  by  $\Gamma(G)$ . We denote by  $C(i) \in \Gamma(G)$  the strongly connected component of  $G$  containing  $i$ . We say that  $C \in \Gamma(G)$  is a *source* if  $C$  has no incoming edges from any vertex not in  $C$ .

► **Lemma 26.** *Let  $\mu$  be an  $(\alpha, \beta)$ -spread distribution on  $\mathcal{D}(n)$  and let  $G_t^\mu$  be its associated graph process. Then, for all  $t \geq 0$ , there is positive probability that the graph  $\Gamma(G_t^\mu)$  has at most*

$$n \cdot (1 - \alpha/4)^t + \frac{2\beta}{\alpha}$$

*sources.*

**Proof.** If  $C \in \Gamma(G)$  is a source, we define the *weight* of  $C$  to be the number of vertices reachable from  $C$  (including vertices of  $C$ ) that are not reachable from any other source of  $G$ . More formally, let

$$\text{Rea}'(C) = \{j \in \text{Rea}(C) \mid j \notin \text{Rea}(C'), \text{ for all sources } C' \in \Gamma(G), C' \neq C\}.$$

Then the weight of a source  $C \in \Gamma(G)$  is denoted by  $w(C) = |\text{Rea}'(C)|$  (we do not define weight for components that are not sources). Let us call a source  $C \in \Gamma(G_t)$  ‘light’ if its weight  $w(C)$  is at most  $k = \frac{\alpha n}{2\beta}$  and ‘heavy’ otherwise. By the definition of weight, there could be at most  $n/k = 2\beta/\alpha$  heavy sources.

We will argue that, in each step, as we move from  $G_t$  to  $G_{t+1}$ , the number of light sources must decrease by a factor of  $(1 - \alpha/4)$  with positive probability. For that purpose, suppose there are  $m_t$  sources in  $G_t$  and among them  $m'_t$  are light. Fix some light source and pick a representative vertex  $i$  from it. Since  $i$  is contained in a light source,  $|\text{Rea}'(C(i))| \leq \frac{\alpha n}{2\beta}$ . When going to  $G_{t+1} = G_t \cup G$ ,  $i$  gets an incoming edge from outside the set  $\text{Rea}'(C(i))$  with probability at least  $\alpha/2$  by Lemma 24. If this happens then in  $G_{t+1}$ , this source will either stop being a source or merge with another source.

Picking a representative for each light source in  $G_t$ , we see that the expected number of representatives  $i$  which get a new incoming edge from outside  $\text{Rea}'(C(i))$  is at least  $(\alpha/2)m'_t$ . Hence, this quantity is obtained with positive probability. Now, if at least  $(\alpha/2)m'_t$  light sources ‘merge’ with another source or stop being a source in  $G_{t+1}$  then the total number of light sources must decrease by at least  $(\alpha/4)m'_t$  (the worst case being that  $(\alpha/4)m'_t$  disjoint pairs of light sources merge with each other). Hence, with positive probability we get that  $m'_{t+1} \leq m'_t \cdot (1 - \alpha/4)$ . Therefore, since the samples in each step  $t$  are independent, there is also a positive probability that  $m'_t \leq n \cdot (1 - \alpha/4)^t$  and  $m_t \leq m'_t + 2\beta/\alpha$ . This completes the proof.  $\blacktriangleleft$

### 3.2 Proof of upper bound from Theorem 9

► **Theorem 27** (Rank of 2-LCSs). *Let  $\mathcal{S}$  be a 2-LCS on  $[n]$  with error-tolerance  $\delta$ . Then  $\text{rank}(\mathcal{S}) \leq O(\frac{1}{\delta} \log_2 n)$ .*

**Proof.** We will work with the (equivalent) set formulation: let  $\mathcal{F} = \{S_1, \dots, S_n\}$  be a set system representing the spanoid  $\mathcal{S}$  as in Lemma 21.

We start by defining an  $(\alpha, \beta)$ -spread distribution  $\mu$  on  $\mathcal{D}(n)$  as follows: To sample a graph  $G$  from  $\mu$  we first pick  $\ell \in [n]$  uniformly at random. Then we add a directed edge from  $j$  to  $i$  for every  $i, j$  such that  $\{j, \ell\} \in M_i$ . In this case we have  $S_j \cap S_\ell \subseteq S_i$  and so, after restricting to  $S_\ell$  we have  $S_j \cap S_\ell \subseteq S_i \cap S_\ell$ .

► **Claim 28.**  *$\mu$  is a  $(2\delta, 1)$ -spread distribution.*

**Proof.** For any fixed  $i \in [n]$ ,  $i$  will get an incoming edge if  $\ell$ , which is randomly chosen from  $[n]$ , belongs to  $M_i$ . Since  $M_i$  has at least  $\delta n$  edges, this will happen with probability at least  $2\delta$ . Now fix any  $i, j \in [n]$ ,  $(j, i)$  will be an edge iff  $\ell$  is equal to the vertex that matches  $j$  in the matching  $M_i$ , this happens with probability at most  $1/n$ . If  $j$  is not matched in  $M_i$ , the probability is zero.  $\blacktriangleleft$

Consider the graph process  $G_t^\mu$  and let  $S_{\ell_1}, \dots, S_{\ell_t}$  be the sets chosen in the  $t$  iterations of sampling from  $\mu$ . If  $i \in \text{Rea}(j)$  in the graph  $G_t^\mu$ , this means that, after restricting to the intersection  $S = S_{\ell_1} \cap \dots \cap S_{\ell_t}$ , the set  $S_j$  is contained in  $S_i$  (i.e.,  $S_j \cap S \subseteq S_i \cap S$ ). By Lemma 26, after  $t = O(\frac{1}{\delta} \log_2 n)$  steps, the graph process  $G_t^\mu$  will contain  $r = O(1/\delta)$  sources. Pick a representative  $S_{a_1}, \dots, S_{a_r}$  from each of these sources. Then, the intersection of the  $t + r = O(\frac{1}{\delta} \log_2 n)$  sets  $S_{\ell_1}, \dots, S_{\ell_t}$  and  $S_{a_1}, \dots, S_{a_r}$  is contained in all  $n$  sets  $S_1, \dots, S_n$ . That is because, when restricted to the intersection of  $S_{\ell_1}, \dots, S_{\ell_t}$ , each set  $S_i$  contains one of the sets  $S_{a_j}, j \in [r]$ .  $\blacktriangleleft$

### 3.3 Proof of upper bound from Theorem 10

► **Theorem 29** (Rank of  $q$ -query LCSs). *Let  $\mathcal{S}$  be a  $q$ -LCS with error-tolerance  $\delta$  and  $q \geq 3$ . Then*

$$\text{rank}(\mathcal{S}) \leq O\left(\delta^{-\frac{1}{q-1}} \cdot n^{\frac{q-2}{q-1}} \log_2 n\right).$$

**Proof.** Like the 2-query case, we work with the set representation  $\mathcal{F} = \{S_1, \dots, S_n\}$  of  $\mathcal{S}$  as in Lemma 21. We follow the same strategy as in the proof of the 2-query case. The difference is that, in this case, we will need to pick many sets to restrict to in each step instead of just one. The first observation is that, if  $\{j_1, \dots, j_q\} \in M_i$  then, restricted to the intersection  $S = S_{j_1} \cap \dots \cap S_{j_{q-1}}$  we have  $S_{j_q} \subset S_i$ . The second observation is that, if we choose a subset  $J \subset [n]$  of size roughly  $n^{\frac{q-2}{q-1}}$  then, in expectation,  $J$  will contain  $q-1$  elements in one of the  $q$ -subsets of  $M_i$  for a constant fraction of the  $i$ 's. Repeating this a logarithmic number of times and using Lemma 26, as in the proof of Theorem 27 will then complete the proof.

We start by defining an  $(\alpha, \beta)$ -spread distribution  $\mu$  on  $\mathcal{D}(n)$ . To sample a graph  $G$  from  $\mu$  we first pick a random set  $J \subset [n]$  such that each  $j \in [n]$  is chosen to be in  $J$  independently with probability  $(\delta n)^{-1/(q-1)}$ . By Markov's inequality we have that

$$\Pr \left[ |J| \geq 4 \cdot \delta^{-\frac{1}{q-1}} n^{\frac{q-2}{q-1}} \right] \leq 1/4. \quad (2)$$

For each  $i \in [n]$  and each  $q$ -subset  $T \in M_i$  we select  $q-1$  elements of  $T$  arbitrarily and refer to them as the *distinguished*  $(q-1)$ -subset of  $T$ . We now argue that, for each  $i \in [n]$ , there is relatively high probability that  $J$  will contain the distinguished  $(q-1)$ -subset of at least one  $q$ -subset in  $M_i$ .

► **Claim 30.** *Let  $E_i$  denote the event that  $J$  contains the distinguished  $(q-1)$ -subset from at least one  $q$ -subset in  $M_i$ . Then, for each  $i \in [n]$  we have that  $\Pr[E_i] \geq 1/2$ .*

**Proof.**  $J$  will contain the distinguished  $q-1$  elements in a specific  $q$ -subset with probability  $(\delta n)^{-1}$ . Since the  $\delta n$   $q$ -subsets in  $M_i$  are disjoint, the probability that  $J$  will not contain any of the distinguished  $(q-1)$ -subsets is at most  $(1 - (1/\delta n))^{\delta n} \leq 1/2$ . ◀

We are now ready to define the edges in the graph  $G$  sampled by  $\mu$ . First we check if  $|J| \geq 4 \cdot \delta^{-\frac{1}{q-1}} n^{\frac{q-2}{q-1}}$ . If this is the case then  $\mu$  outputs the empty graph (by Eq.2 this happens with probability at most  $1/4$ ). Otherwise for each  $i \in [n]$  we check to see if  $J$  contains the distinguished  $(q-1)$ -subset from one of the  $q$ -subsets of  $M_i$ . If there is at least one such  $q$ -subset, we pick one of them uniformly at random. Suppose the  $q$ -subset we chose is  $\{j_1, \dots, j_q\}$  and that the distinguished elements are the first  $q-1$ . Then we add the directed edge  $j_q \rightarrow i$  to the graph  $G$ . By the above discussion, we know that, restricted to the intersection of all sets indexed by  $J$  the set  $S_{j_q}$  is contained in  $S_i$  (hence the directed edge representing set inclusion).

► **Claim 31.**  *$\mu$  is  $(1/4, 1/\delta)$ -spread.*

**Proof.** By Claim 30, and since the probability that  $|J|$  is too large is at most  $1/4$  we see that any fixed  $i \in [n]$  will get an incoming edge in  $G$  with probability at least  $\alpha = 1/4$ . For any fixed  $i, j \in [n]$ , since the distribution of the special  $q$ -subset which is contributing an edge to  $i$  is uniform in  $M_i$  (conditioned on  $J$  containing a  $q$ -subset from  $M_i$ ), we can conclude that  $(j, i) \in E(G)$  with probability at most  $1/(\delta n) = \beta/n$ . This proves the claim. ◀

Now, applying Lemma 26, we get that, after  $t = O(\log_2 n)$  steps, the graph process  $G_t^\mu$  will contain at most  $O(1/\delta)$  sources with positive probability. Let  $J_1, \dots, J_t$  be the sets chosen in the different steps of the process and, w.l.o.g, remove any of them that were too big (i.e., when the graph sampled by  $\mu$  was empty). Hence, all of the sets satisfy  $|J_i| \leq 4 \cdot \delta^{-\frac{1}{q-1}} n^{\frac{q-2}{q-1}}$ . Now, let  $S$  be the intersection of all sets  $S_j$  such that  $j$  belongs to at least one of the sets  $J_i$ .

Then, restricted to  $S$ , each of the sets  $S_i$  contains one of the sources in the graph  $G_t^\mu$ . Hence, if we add to our intersection a representative from each of the sources, we will get a set that is contained in all the sets  $S_j$ . The total number of sets we end up intersecting is bounded by

$$O(1/\delta) + \sum_{i=1}^t |J_i| = O\left(\delta^{-\frac{1}{q-1}} \cdot n^{\frac{q-2}{q-1}} \log_2 n\right).$$

This completes the proof of the theorem.  $\blacktriangleleft$

#### 4 Constructing $q$ -LCSs with high rank

In this section we prove the lower bound part of Theorem 10 (the lower bound for the 2-query case follows from the Hadamard code construction). We will in fact generate this spanoid at random by picking, for each  $i \in [n]$ , a random  $q$ -matching  $M_i$  on  $[n]$  and, for each  $q$ -subset  $T \in M_i$  add the rule  $T \models i$ . The resulting spanoid will thus have, by design, the structure of a  $q$ -LCS. The reason why this spanoid should have high rank (with high probability) relies on the following observation. Suppose  $A \subset [n]$  is a set that spans  $[n]$ . This means that there is a sequence of derivations  $T_i \models i$  with each  $q$ -subset  $T_i$  in the matching  $M_i$  that eventually generates all of  $[n]$ . We can limit ourselves to the first  $C \cdot |A|$  such derivations for some large  $C$ . These derivations generate a set  $A'$  of size  $(C+1)|A|$  (including the original  $A$  and the  $C|A|$  newly derived elements). Now, the set  $A'$  must contain all of the  $q$ -subsets  $T_i$  for  $C|A|$  values of  $i$ . However, the union of randomly chosen  $C|A|$   $q$ -subsets will generally have size much larger than  $(C+1)|A|$  (closer to  $q \cdot C|A|$ ).

**► Theorem 32** (Existence of high rank  $q$ -LCSs). *For any integer  $q \geq 3$  and all sufficiently large  $n$  the following holds. Consider the following distribution generating a spanoid  $\mathcal{S}$  on base set  $[n]$ . For each  $i \in [n]$  pick a  $q$ -matching  $M_i$  of size  $\lfloor n/2q \rfloor$  uniformly at random and add the rule  $T \models i$  for all  $T \in M_i$ . Then, with probability approaching one,  $\text{rank}(\mathcal{S})$  is larger than  $r = cn^{\frac{q-1}{q-2}} / \log_2(n)$ , where  $0 < c < 1$  is an absolute constant.*

**Proof.** Let  $m = r \cdot \log_2(n) = cn^{\frac{q-1}{q-2}}$ . If the rank of  $\mathcal{S}$  is at most  $r$  then there exists a set  $A \subset [n]$  of size  $r$  that spans (using the rules obtained from the  $n$  random matchings  $M_1, \dots, M_n$ ) the entire base set  $[n]$ . We will upper bound the probability that such a set exists by bounding the smaller event given by the existence of a set of  $m$  rules that can be applied one after another starting with the original set  $A$ . That is, let  $\mathcal{E}$  denote the event that there exists a set  $A$  of size  $r$  on which one can sequentially apply  $m$  rules of the form  $T_{j_i} \models j_i$  with each  $T_{j_i}$  belonging to the matching  $M_{j_i}$  and for  $m$  different values  $j_1, \dots, j_m \in [n]$  arriving at the final set  $\hat{A} = A \cup \{j_1, \dots, j_m\}$ . If  $A$  spans  $[n]$  then clearly the event  $\mathcal{E}$  must hold and so, it is enough to show that  $\mathcal{E}$  happens with probability approaching zero.

We will present the event  $\mathcal{E}$  as the union of (possibly overlapping) smaller events and then use the union bound, bounding the probability that each one occurs and multiplying by the number of bad events. Given a set  $A \subset [n]$  of size  $r$ , a tuple of  $m$  indices  $\hat{J} = \{j_1, j_2, \dots, j_m\}$  and a family of  $q$ -subsets  $\hat{T} = \{T_{j_1}, \dots, T_{j_m}\}$  with  $T_{j_i} \in M_{j_i}$  denote by  $\mathcal{E}(A, \hat{J}, \hat{T})$  the event in which the set  $A$  spans the set  $\hat{A} = A \cup \hat{J}$  using the rules  $T_{j_i} \models j_i$  applied in order with  $i$  going from 1 to  $m$ . For every fixing of  $A, \hat{J}, \hat{T}$  we can bound

$$\Pr[\mathcal{E}(A, \hat{J}, \hat{T})] \leq \prod_{i=1}^m \Pr[T_{j_i} \subset \hat{A}].$$

W.l.o.g suppose we sample the random matchings iteratively, picking a new  $q$ -subset at random among the available elements not covered by any previously chosen  $q$ -subsets in the current matching. Since the number of  $q$ -subsets in each matching is  $\lfloor n/2q \rfloor$  we have, at each step, at least  $n/2$  available elements to chose from and so

$$\Pr[T_{j_i} \subset \hat{A}] \leq \frac{\binom{m+r}{q}}{\binom{n/2}{q}} \leq \left(\frac{4m}{n}\right)^q.$$

Taking the product over all  $m$   $q$ -subsets in  $\hat{T}$  we get

$$\Pr[\mathcal{E}(A, \hat{J}, \hat{T})] \leq \left(\frac{4m}{n}\right)^{qm}.$$

To complete the proof we bound the number of tuples  $(A, \hat{J}, \hat{T})$  as above by

$$\binom{n}{r} \cdot \binom{n}{m} \cdot \lfloor n/2q \rfloor^m \leq n^r \cdot (en/m)^m \cdot n^m \leq \left(\frac{6n^2}{m}\right)^m,$$

where the last inequality used the fact that  $r/m \leq 1/\log_2(n)$ . Putting these bounds together we get that

$$\Pr[\mathcal{E}] \leq \left(\frac{4m}{n}\right)^{qm} \left(\frac{6n^2}{m}\right)^m = \left(\frac{6 \cdot 4^q \cdot m^{q-1}}{n^{q-2}}\right)^m$$

which is exponentially decreasing in  $m$  for the given choice of  $m = c \cdot n^{(q-2)/(q-1)}$  and for  $c$  a sufficiently small constant.  $\blacktriangleleft$

One could ask for a more explicit construction of an LCS with rank equal to (or even close to) that stated above. We are not able to give such a construction but can relate this problem to a longstanding open problem in explicit construction of expander graphs. A bipartite (balanced) expander of degree  $q$ , is a bipartite graph with  $n$  left vertices  $L$  and  $n$  right vertices  $R$  such that the degree of each vertex is  $q$  and such that sets  $A \subset L$  of size ‘not too large’ have many neighbors in  $R$ . More specifically, one typically asks that sets with  $|A| \leq n/2$  have at least  $(1 + \epsilon)|A|$  right neighbors for some constant  $\epsilon > 0$ . It is quite easy to see that a random graph of this form will be a good expander with high probability and, by now, there are also many explicit constructions [17]. One can also consider *unbalanced* bipartite expanders in which  $|L| \gg |R|$ . Take, for example, the setting in which  $|L| = n^2$ ,  $|R| = n$  and when the degree of every vertex in  $L$  is some constant  $q$ . A simple probabilistic argument shows that sufficiently small sets in  $L$ , namely sets of size  $|A| \leq n^{\alpha_q}$  with  $\alpha_q < 1$  a constant depending on  $q$  and approaching 1 as  $q$  grows, have many neighbors in  $R$  (say, at least  $2|A|$ ). However, no explicit constructions of such graphs are known (for any constant  $q$  and any  $\alpha_q > 0$ ). The property we needed in our random construction of LCSs can be thought of as an ‘easier’ variant of the expander construction problem. Given  $q$ -matchings  $M_1, \dots, M_n$  each of size  $\delta n$  consider the bipartite graph with  $L = [n] \times [\delta n]$  and  $R = [n]$ . We identify each vertex  $(i, j) \in L$  with the  $j$ ’th  $q$ -subset  $T_{ij}$  of  $M_i$  and connect it to the  $q$  neighbors in  $R$  given by that  $q$ -subset. For our proof to work we need the property that there is no small set containing many  $q$ -subsets from different matchings. This corresponds to asking for the above graph to be an expander for a restricted family of sets, namely to sets that have at most one vertex  $(i, j)$  for a given  $i$  (with each subgraph  $(i, *)$  defining a matching).

## 5 Conclusion and open problems

Our work introduces the abstract notion of a spanoid in the hope that further study of its properties will lead to progress on LCCs and perhaps in other areas. We list below some concrete directions for future work.

1. We showed that there exist spanoids, called  $q$ -LCSs, which “look like”  $q$ -LCCs and whose rank matches the best known upper bounds. Can we bypass this ‘barrier’ by using additional properties of LCCs? We have at least two examples where this was possible. One is the result of [21] for LCCs over constant size alphabet and the other is the work in [11] for linear 3-LCCs over the real numbers. The bounds of [21] crucially depend on the alphabet having small size and the bounds in [11] exploit properties of real numbers.
2. Understanding the possible gap between functional rank and formal rank of a spanoid is a very interesting question. We proved that there can be a polynomial gap. The next challenge is to find a spanoid on  $n$  elements whose f-rank is  $n^{o(1)}$  and rank is  $n^{\Omega(1)}$ . Naturally,  $q$ -LCSs for constant  $q \geq 3$  are plausible candidates for this. If there are no such spanoids, then it would imply the existence of  $q$ -LCCs of length  $n$  and  $n^{\Omega_q(1)}$  dimension!<sup>1</sup>
3. Suppose we start with a functional representation with large alphabet, can we do alphabet reduction without losing too many codewords?
4. We have seen that one way to go past the rank barrier is to use  $\text{LP}^{\text{entropy}}$ . Can we improve the existing upper bounds on the dimension of  $q$ -LCCs by upper bounding  $\text{LP}^{\text{entropy}}$  of  $q$ -LCSs? Can we use LP duality and construct good feasible solutions to the dual of  $\text{LP}^{\text{entropy}}$  to prove good upper bounds on  $\text{LP}^{\text{entropy}}$ ?
5. What are other connections of spanoids to existing theory of set systems, matroids, algebraic equations and other problems described in the introduction?

---

### References

- 1 Michael Alekhnovich, Eli Ben-Sasson, Alexander A Razborov, and Avi Wigderson. Pseudorandom generators in propositional proof complexity. *SIAM Journal on Computing*, 34(1):67–88, 2004.
- 2 Michael Alekhnovich and Alexander A Razborov. Lower bounds for polynomial calculus: Non-binomial case. In *Foundations of Computer Science, 2001. Proceedings. 42nd IEEE Symposium on*, pages 190–199. IEEE, 2001.
- 3 Boaz Barak, Zeev Dvir, Avi Wigderson, and Amir Yehudayoff. Fractional Sylvester-Gallai theorems. *Proceedings of the National Academy of Sciences*, 2012.
- 4 Boaz Barak, Zeev Dvir, Amir Yehudayoff, and Avi Wigderson. Rank bounds for design matrices with applications to combinatorial geometry and locally correctable codes. In *Proceedings of the forty-third annual ACM symposium on Theory of computing*, pages 519–528. ACM, 2011.
- 5 Eli Ben-Sasson and Avi Wigderson. Short proofs are narrow—resolution made simple. *Journal of the ACM (JACM)*, 48(2):149–169, 2001.
- 6 Arnab Bhattacharyya, Sivakanth Gopi, and Avishay Tal. Lower Bounds for 2-Query LCCs over Large Alphabet. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, APPROX/RANDOM 2017, August 16-18, 2017, Berkeley, CA, USA*, pages 30:1–30:20, 2017. doi:10.4230/LIPIcs.APPROX-RANDOM.2017.30.
- 7 Béla Bollobás. Weakly  $k$ -saturated graphs. *Beiträge zur Graphentheorie (Kolloquium, Manebach, 1967)*, Teubner, Leipzig, pages 25–31, 1968.

---

<sup>1</sup> Possibly over a large alphabet.



- 8 Henning Bruhn and Oliver Schaudt. The journey of the union-closed sets conjecture. *Graphs and Combinatorics*, 31(6):2043–2074, 2015.
- 9 Irit Dinur and Tali Kaufman. Dense locally testable codes cannot have constant rate and distance. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*, pages 507–518. Springer, 2011.
- 10 Zeev Dvir. Incidence Theorems and Their Applications. *Foundations and Trends® in Theoretical Computer Science*, 6(4):257–393, 2012. doi:10.1561/04000000056.
- 11 Zeev Dvir, Shubhangi Saraf, and Avi Wigderson. Breaking the quadratic barrier for 3-LCC’s over the reals. In *Proceedings of the 46th Annual ACM Symposium on Theory of Computing*, pages 784–793. ACM, 2014.
- 12 Klim Efremenko. 3-query locally decodable codes of subexponential length. In *STOC*, pages 39–44, 2009. doi:10.1145/1536414.1536422.
- 13 Paul Erdős, Peter Frankl, and Zoltán Füredi. Families of finite sets in which no set is covered by the union of others. *Israel Journal of Mathematics*, 51(1):79–89, 1985.
- 14 Jacob Fox, Choongbum Lee, and Benny Sudakov. Maximum union-free subfamilies. *Israel Journal of Mathematics*, 191(2):959–971, 2012.
- 15 Zoltán Füredi. Onr-Cover-free Families. *J. Comb. Theory Ser. A*, 73(1):172–173, January 1996. doi:10.1006/jcta.1996.0012.
- 16 Lianna Hambardzumyan, Hamed Hatami, and Yingjie Qian. Polynomial method and graph bootstrap percolation. *arXiv preprint*, 2017. arXiv:1708.04640.
- 17 Shlomo Hoory, Nathan Linial, and Avi Wigderson. Expander graphs and their applications. *Bulletin of the American Mathematical Society*, 43(4):439–561, 2006.
- 18 Eran Iceland and Alex Samorodnitsky. On coset leader graphs of structured linear codes. *arXiv preprint*, 2018. arXiv:1802.01184.
- 19 Jonathan Katz and Luca Trevisan. On the efficiency of local decoding procedures for error-correcting codes. In *Proceedings of the 32nd annual ACM symposium on Theory of computing (STOC 2000)*, pages 80–86. ACM Press, 2000.
- 20 Leroy Milton Kelly. A resolution of the Sylvester-Gallai problem of J.-P. Serre. *Discrete & Computational Geometry*, 1(2):101–104, 1986.
- 21 Iordanis Kerenidis and Ronald de Wolf. Exponential lower bound for 2-query locally decodable codes via a quantum argument. *J. of Computer and System Sciences*, 69:395–420, 2004. Preliminary version appeared in STOC’03.
- 22 Emanuel Knill. Graph generated union-closed families of sets. *arXiv preprint*, 1994. arXiv:math/9409215.
- 23 Amir Shpilka. Sylvester-Gallai type theorems for quadratic polynomials. *Private communication*, 2018.
- 24 Piotr Wójcik. Union-closed families of sets. *Discrete Mathematics*, 199(1-3):173–182, 1999.
- 25 David Woodruff. New Lower Bounds for General Locally Decodable Codes. *Electronic Colloquium on Computational Complexity (ECCC)*, 14(006), 2007.
- 26 Sergey Yekhanin. Towards 3-query locally decodable codes of subexponential length. *Journal of the ACM (JACM)*, 55(1):1, 2008.
- 27 Sergey Yekhanin. Locally decodable codes. *Foundations and Trends® in Theoretical Computer Science*, 6(3):139–255, 2012.

# Fairness Under Composition

Cynthia Dwork<sup>1</sup>

Harvard John A Paulson School of Engineering and Applied Science,  
Radcliffe Institute for Advanced Study, Cambridge, MA, USA  
dwork@seas.harvard.edu

Christina Ilvento<sup>2</sup>

Harvard John A Paulson School of Engineering and Applied Science,  
Cambridge, MA, USA  
cilvento@g.harvard.edu

---

## Abstract

Algorithmic fairness, and in particular the fairness of scoring and classification algorithms, has become a topic of increasing social concern and has recently witnessed an explosion of research in theoretical computer science, machine learning, statistics, the social sciences, and law. Much of the literature considers the case of a single classifier (or scoring function) used once, in isolation. In this work, we initiate the study of the fairness properties of systems composed of algorithms that are fair in isolation; that is, we study *fairness under composition*. We identify pitfalls of naïve composition and give general constructions for fair composition, demonstrating both that classifiers that are fair in isolation do not necessarily compose into fair systems and also that seemingly unfair components may be carefully combined to construct fair systems. We focus primarily on the individual fairness setting proposed in [Dwork, Hardt, Pitassi, Reingold, Zemel, 2011], but also extend our results to a large class of group fairness definitions popular in the recent literature, exhibiting several cases in which group fairness definitions give misleading signals under composition.

**2012 ACM Subject Classification** Theory of computation → Computational complexity and cryptography, Theory of computation → Design and analysis of algorithms, Theory of computation → Theory and algorithms for application domains

**Keywords and phrases** algorithmic fairness, fairness, fairness under composition

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2019.33

**Related Version** A full version of the paper is available at <https://arxiv.org/abs/1806.06122>.

## 1 Introduction

As automated decision-making extends its reach ever more deeply into our lives, there is increasing concern that such decisions be fair. The rigorous theoretical study of fairness in algorithmic classification was initiated by Dwork *et al* in [4] and subsequent works investigating alternative definitions, fair representations, and impossibility results have proliferated in the machine learning, economics and theoretical computer science literatures.<sup>3</sup> The notions of fairness broadly divide into *individual fairness*, requiring that individuals who are similar with respect to a given classification task (as measured by a task-specific

---

<sup>1</sup> This work was supported in part by Microsoft Research and the Sloan Foundation.

<sup>2</sup> This work was supported in part by the Smith Family Fellowship and Microsoft Research.

<sup>3</sup> See also [20] [9] and [10], which predate [4] and are motivated by similar concerns.



similarity metric) have similar probability distributions on classification outcomes; and *group fairness*, which requires that different demographic groups experience the same treatment in some average sense.

In a bit more detail, a *classification task* is the problem of mapping *individuals* to *outcomes*; for example, a decision task may map individuals to outcomes in  $\{0, 1\}$ . A classifier is a possibly randomized algorithm solving a classification task. In this work we initiate the study of *fairness under composition*: what are the fairness properties of systems built from classifiers that are fair in isolation? Under what circumstances can we ensure fairness, and how can we do so? A running example in this work is online advertising. If a set of advertisers, say, one for tech jobs and one for a grocery delivery service, compete for the attention of users, say one for tech jobs and one for a grocery delivery service, and each chooses fairly whether to bid (or not), is it the case that the advertising system, including budget handling and tie-breaking, will also be fair?

We identify and examine several types of composition and draw conclusions about auditing systems for fairness, constructing fair systems, and definitions of fairness for systems. In the remainder of this section we summarize our results and discuss related work. A full version of this paper, containing complete proofs of all our results, is available on ArXiv.

### Task-Competitive Compositions

We first consider the problem of two or more tasks competing for individuals, motivated by the online advertising setting described above. We prove that two advertisers for different tasks, each behaving fairly (when considered independently), will not necessarily produce fair outcomes when they compete. Intuitively (and as empirically observed by [17]), the attention of individuals similarly qualified for a job may effectively have different costs due to these individuals' respective desirability for other advertising tasks, like household goods purchases. That is, individuals claimed by the household goods advertiser will not see the jobs ad, regardless of their job qualification. These results are not specific to an auction setting and are robust to choice of “tie-breaking” functions that select among multiple competing tasks (advertisers). Nonetheless, we give a simple mechanism, `RandomizeThenClassify`, that solves the fair task-competitive classification problem using classifiers for the competing tasks, each of which is fair in isolation, in a black-box fashion and without modification. In the Appendix (Lemma 15) we give a second technique for modifying the fair classifier of the lower bidder (loser of the tie-breaking function) in order to achieve fairness.

### Functional Compositions

Is the “OR” of two fair classifiers also fair? More generally, when can we build fair classifiers by computing on values that were fairly obtained? Here we must understand what is the salient outcome of the computation. For example, when reasoning about whether the college admissions system is fair, the salient outcome may be whether a student is accepted to at least one college, and not whether the student is accepted to a specific college<sup>4</sup>. Even if each college uses a fair classifier, the question is whether the “OR” of the colleges' decisions is fair. Furthermore, an acceptance to college may not be meaningful without sufficient accompanying financial aid. Thus in practice, we must reason about the OR of ANDs of acceptance and financial aid across many colleges. We show that although in general there

---

<sup>4</sup> In this simple example, we assume that all colleges are equally desirable, but it is not difficult to extend the logic to different sets of comparable colleges.

are no guarantees on the fairness of functional compositions of fair components, there are some cases where fairness in ORs can be satisfied. Such reasoning can be used in many applications where long-term and short-term measures of fairness must be balanced. In the case of feedback loops, where prior positive outcomes can improve the chances of future positive outcomes, functional composition provides a valuable tool for determining at which point(s) fairness must be maintained and determining whether the existing set of decision procedures will adhere to these requirements.

### Dependent Compositions

There are many settings in which each individual's classifications are dependent on the classifications of others. For example, if a company is interviewing a set of job candidates in a particular order, accepting a candidate near the beginning of the list precludes any subsequent candidates from even being considered. Thus, even if each candidate actually considered is considered fairly in isolation, dependence between candidates can result in highly unfair outcomes. For example, individuals who are socially connected to the company through friends or family are likely to hear about job openings first and thus be considered for a position before candidates without connections. We show that selecting a cohort of people – online or offline – requires care to prevent dependencies from undermining an independently fair selection mechanism. We address this in the offline case with two randomized constructions, `PermuteThenClassify` and `WeightedSampling`. These algorithms can be applied in the online case, even under adversarial ordering, provided the size of the universe of individuals is known; when this is not known there is no solution.

### Nuances of group-based definitions

Many fairness definitions in the literature seek to provide fairness guarantees based on group-level statistical properties. For example, “Equal Opportunity” [6] requires that, conditioned on qualification, the probability of a positive outcome is independent of protected attributes such as race or gender. Group Fairness definitions have practical appeal in that they are possible to measure and enforce empirically without reference to a task-specific similarity metric.<sup>5</sup> We extend our results to group fairness definitions and we also show that these definitions do not always yield consistent signals under composition. In particular, we show that the intersectional subgroup concerns (which motivate [11, 7]) are exacerbated by composition. For example, an employer who uses group fairness definitions to ensure parity with respect to race and gender may fail to identify that “parents” of particular race and gender combinations are not treated fairly. Task-competitive composition exacerbates this problem, as the employers may be prohibited from even collecting parental status information, but their hiring processes may be composed with other systems which legitimately differentiate based on parental status.

Finally, we also show how naïve strategies to mitigate these issues in composition may result in learning a nominally fair solution that is clearly discriminating against a socially meaningful subgroup not officially called out as “protected,” from which we conclude that understanding the behavior of fairness definitions under composition is critical for choosing which definition is meaningful in a given setting.

---

<sup>5</sup> However, defining and measuring qualification may require care.

## Implications of Our Results

Our composition results have several practical implications. First, testing individual components without understanding of the whole system will be insufficient to safely draw either positive or negative conclusions about the fairness of the system. Second, composition properties are an important point of evaluation for any definitions of fairness or fairness requirements imposed by law or otherwise. Failing to take composition into account when specifying a group-based fairness definition may result in a meaningless signal under composition, or worse may lead to ingraining poor outcomes for certain subgroups while still nominally satisfying fairness requirements. Third, understanding of the salient outcomes on which to measure and enforce fairness is critical to building meaningfully fair systems. Finally, we conclude that there is significant potential for improvement in the mechanisms proposed for fair composition and many settings in which new mechanisms could be proposed.

### 1.1 Related Work

Fairness retained under post-processing in the single-task one-shot setting is central in [22, 19, 4]. The definition of individual fairness we build upon in this work was introduced by Dwork *et al.* in [4]. Learning with oracle access to the fairness metric is considered by [5, 13]. A number of group-based fairness definitions have been proposed, and Ritov *et al.* provide a combined discussion of the parity-based definitions in [21]. In particular, their work includes discussion of Hardt *et al.*'s Equality of Opportunity and Equal Odds definitions and Kilbertus *et al.*'s Counterfactual Fairness [6, 12]. Kleinberg *et al.* and Chouldechova independently described several impossibility results related to simultaneously satisfying multiple group fairness conditions in single classification settings [14],[2].

Two concurrent lines of work aiming to bridge the gap between individual and group consider ensuring fairness properties for large numbers of large groups and their (sufficiently large) intersections [11, 7]. While these works consider the one-shot, single-task setting, we will see that group intersection properties are of particular importance under composition. Two subsequent works in this general vein explore approximating individual fairness with the help of an oracle that knows the task-specific metric [13, 5]. Two works also consider how feedback loops can influence fair classification, and how interventions can help [8, 18].

Several empirical or observational studies document the effects of multiple task composition. For example, Lambrecht and Tucker study how intended gender-neutral advertising can result in uneven delivery due to high demand for the attention of certain demographics [17]. Datta *et al.* also document differences in advertising based on gender, although they are agnostic as to whether the cause is due to multiple task composition or discriminatory behavior on the part of the advertisers or platform [3]. Whether it is truly “fair” that, say, home goods advertisers bid more highly for the attention of women than for the attention of men, may be debatable, although there are clearly instances in which differential targeting is justified, such as when advertising maternity clothes. This *actuarial fairness* is the industry practice, so we pose a number of examples in this framework and analyze the implications of composition.

## 2 Preliminary Definitions and Assumptions

### 2.1 General Terminology

We refer to classifiers as being “fair in isolation” or “independently fair” to indicate that with no composition, the classifier satisfies a particular fairness definition. In such cases expectation and probability are taken over the randomness of the classification procedure

and, for group fairness, selection of elements from the universe. We denote the universe of individuals relevant for a task as  $U$ , and we generally use  $u, v, w \in U$  to refer to universe elements. We generally consider binary classifiers in this work, and use  $p_w$  to denote the probability of assigning the positive outcome (or simply 1) to the element  $w$  for a particular classifier. We generally write  $C : U \times \{0, 1\}^* \rightarrow \{0, 1\}$ , where  $\{0, 1\}^*$  represents the random bits of the classifier. This allows us to comfortably express the probability of positive classification  $\mathbb{E}_r[C(u)]$  as well as the output of the classifier under particular randomness  $C(u, r)$ . In this notation,  $p_u = \mathbb{E}_r[C(u)]$ . When considering the distribution on outputs of a classifier  $C$ , we use  $\tilde{C} : U \rightarrow \Delta(\{0, 1\})$ . When two or more classifiers or tasks are compared, we either use a subscript  $i$  to indicate the  $i^{\text{th}}$  classifier or task, or a prime ( $'$ ) to indicate the second classifier or task. For example  $\{C, C'\}$ ,  $\{C_i | i \in [k]\}$ ,  $\{T, T'\}$ ,  $\{T_i | i \in [k]\}$ .

## 2.2 Individual Fairness

Throughout this work, our primary focus is on *individual fairness*, proposed by Dwork *et al* in [4]. As noted above, a *classification task* is the problem of mapping *individuals* in a universe to *outcomes*.

► **Definition 1** (Individual Fairness [4]). Let  $d : \Delta(O) \times \Delta(O) \rightarrow [0, 1]$  denote the total variation distance on distributions over  $O$ <sup>6</sup>. Given a universe of individuals  $U$ , and a task-specific metric  $\mathcal{D}$  for a classification task  $T$  with outcome set  $O$ , a randomized classifier  $C : U \times \{0, 1\}^* \rightarrow O$ , such that  $\tilde{C} : U \rightarrow \Delta(O)$ , is *individually fair* if and only if for all  $u, v \in U$ ,  $\mathcal{D}(u, v) \geq d(\tilde{C}(u), \tilde{C}(v))$ .

Note that when  $|O| = 2$  we have  $d(\tilde{C}(u), \tilde{C}(v)) = |\mathbb{E}_r[C(u)] - \mathbb{E}_r[C(v)]| = |p_u - p_v|$ . In several proofs we will rely on the fact that it is possible to construct individually fair classifiers with particular distance properties (see Lemma 16 and corollaries in the Appendix).

## 2.3 Group Fairness

In principle, all our individual fairness results extend to group fairness definitions; however, there are a number of technicalities and issues unique to group fairness definitions, which we discuss in Section 6. Group fairness is often framed in terms of *protected attributes*  $\mathcal{A}$ , such as sex, race, or socio-economic status, while allowing for differing treatment based on a set of *qualifications*  $\mathcal{Z}$ , such as, in the case of advertising, the willingness to buy an item. Conditional Parity, a general framework proposed in [21] for discussing these definitions, conveniently captures many of the popular group fairness definitions popular in the literature including Equal Odds and Equal Opportunity [6], and Counterfactual Fairness [16].

► **Definition 2** (Conditional Parity [21]). A random variable  $\mathbf{x}$  satisfies parity with respect to  $\mathbf{a}$  conditioned on  $\mathbf{z} = z$  if the distribution of  $\mathbf{x} \mid (\mathbf{a}, \{\mathbf{z} = z\})$  is constant in  $\mathbf{a}$ :  $\Pr[\mathbf{x} = x \mid (\mathbf{a} = a, \mathbf{z} = z)] = \Pr[\mathbf{x} = x \mid (\mathbf{a} = a', \mathbf{z} = z)]$  for any  $a, a' \in \mathcal{A}$ . Similarly,  $\mathbf{x}$  satisfies parity with respect to  $\mathbf{a}$  conditioned on  $\mathbf{z}$  (without specifying a value of  $\mathbf{z}$ ) if it satisfies parity with respect to  $\mathbf{a}$  conditioned on  $\mathbf{z} = z$  for all  $z \in \mathcal{Z}$ . All probabilities are over the randomness of the prediction procedure and the selection of elements from the universe.

<sup>6</sup> [4] also considered other notions of distributional distance.

### 3 Multiple-Task Composition

First, we consider the problem of composition of classifiers for multiple tasks where the outcome for more than one task is decided. Multiple Task Fairness, defined next, requires fairness to be enforced independently and simultaneously for each task.

► **Definition 3** (Multiple Task Fairness). For a set  $\mathcal{T}$  of  $k$  tasks with metrics  $\mathcal{D}_1, \dots, \mathcal{D}_k$ , a (possibly randomized) system  $\mathcal{S} : U \times r \rightarrow \{0, 1\}^k$ , which assigns outputs for task  $i$  in the  $i^{\text{th}}$  coordinate of the output, satisfies multiple task fairness if for all  $i \in [k]$  and all  $u, v \in U$   $\mathcal{D}_i(u, v) \geq |\mathbb{E}[\mathcal{S}_i(u)] - \mathbb{E}[\mathcal{S}_i(v)]|$  where  $\mathbb{E}[\mathcal{S}_i(u)]$  is the expected outcome for the  $i^{\text{th}}$  task in the system  $\mathcal{S}$  and where the expectation is over the randomness of the system and all its components.

#### 3.1 Task-Competitive Composition

We now pose the relevant problem for multiple task fairness: competitive composition.

► **Definition 4** (Single Slot Composition Problem). A (possibly randomized) system  $\mathcal{S}$  is said to be a solution to the single slot composition problem for a set of  $k$  tasks  $\mathcal{T}$  with metrics  $\mathcal{D}_1, \dots, \mathcal{D}_k$ , if  $\forall u \in U$ ,  $\mathcal{S}$  assigns outputs for each task  $\{x_{u,1}, \dots, x_{u,k}\} \in \{0, 1\}^k$  such that  $\sum_{i \in [k]} x_{u,i} \leq 1$ , and  $\forall i \in [k]$ , and  $\forall u, v \in U$ ,  $\mathcal{D}_i(u, v) \geq |\mathbb{E}[x_{u,i}] - \mathbb{E}[x_{v,i}]|$ .

The single slot composition problem captures the scenario in which an advertising platform may have a single slot to show an ad but need not show any ad. Imagine that this advertising system only has two types of ads: those for jobs and those for household goods. If a person is qualified for jobs and eager and able to purchase household goods, the system must pick at most one of the ads to show. In this scenario, it may be unlikely that the advertising system would choose to show no ads, but the problem specification does not require that any positive outcome is chosen.

To solve the single-slot composition problem we must build a system which chooses at most one of the possible tasks so that fairness is preserved simultaneously for each task, across all elements in the universe. Clearly if classifiers for each task may *independently* and fairly assign outputs without interference, the system as a whole satisfies multiple task fairness. However, most systems will require trade-offs between tasks. Consider a naïve solution to the single-slot problem for ads: each advertiser chooses to bid on each person with some probability, and if both advertisers bid for the same person, the advertiser with the higher bid gets to show her ad. Formally, we define a tie-breaking function and Task-Competitive Composition:

► **Definition 5** (Tie-breaking Function). A (possibly randomized) *tie-breaking function*  $\mathbb{B} : U \times \{0, 1\}^* \times \{0, 1\}^k \rightarrow [k] \cup \{0\}$  takes as input an individual  $w \in U$  and a  $k$ -bit string  $x_w$  and outputs the index of a “1” in  $x_w$  if such an index exists and 0 otherwise.

For notational convenience, in the case of two tasks  $T$  and  $T'$ , we use  $\mathbb{B}_w(T)$  to refer to the probability that  $\mathbb{B}$  chooses task  $T$  for element  $w$  if both  $T$  and  $T'$  return positive classifications, and analogously define  $\mathbb{B}_w(T')$ .

► **Definition 6** (Task-Competitive Composition). Consider a set  $\mathcal{T}$  of  $k$  tasks, and a tie-breaking function as defined above. Given a set  $\mathcal{C}$  of classifiers for the set of tasks, define  $y_w = \{y_{w,1}, \dots, y_{w,k}\}$  where  $y_{w,i} = C_i(w)$ . The task-competitive composition of the set  $\mathcal{C}$  is defined as  $y_w^* = \mathbb{B}(w, y_w)$  for all  $w \in U$ .



Definition 6 yields a system  $S$  defined by  $S(w) = 0^k$  if  $y_w = 0^k$  and  $S(w) = e_{\mathbb{B}(w, y_w)}$  (the  $\mathbb{B}(w, y_w)$  basis vector of dimension  $k$ ) if  $y_w \neq 0^k$ . We evaluate its fairness by examining the Lipschitz requirements  $|\Pr[y_u^* = i] - \Pr[y_v^* = i]| \leq \mathcal{D}_i(u, v)$  for all  $u, v \in U$  and  $i \in [k]$ .

Task-competitive composition can reflect many scenarios other than advertising, which are discussed in greater detail in the full paper. Note that the tie-breaking function need not encode the same logic for all individuals and may be randomized. We start by introducing Lemma 7, which handles the simple case for a strict tie-breaking function for all individuals, and extend to all tie-breaking functions in Theorem 8.

► **Lemma 7.** *For any two tasks  $T$  and  $T'$  such that the metrics for each task ( $\mathcal{D}$  and  $\mathcal{D}'$  respectively) are not identical and are non-trivial<sup>7</sup> on a universe  $U$ , and if there is a strict preference for  $T$ , that is  $\mathbb{B}_w(T) = 1 \forall w \in U$ , then there exists a pair of classifiers  $\mathcal{C} = \{C, C'\}$  which are individually fair in isolation but when combined with task-competitive composition violate multiple task fairness.*

**Proof.** We construct a pair of classifiers  $\mathcal{C} = \{C, C'\}$  which are individually fair in isolation for the tasks  $T$  and  $T'$ , but do not satisfy multiple task fairness when combined with task-competitive composition with a strict preference for  $T$  for all  $w \in U$ . Task-competitive composition ensures that at most one task can be classified positively for each element, so our strategy is to construct  $C$  and  $C'$  such that the distance between a pair of individuals is stretched for the ‘second’ task.

By non-triviality of  $\mathcal{D}$ , there exist  $u, v$  such that  $\mathcal{D}(u, v) \neq 0$ . Fix such a pair  $u, v$  and let  $p_u$  denote the probability that  $C$  assigns 1 to  $u$ , and analogously  $p_v, p'_u, p'_v$ . We use these values as placeholders, and show how to set them to prove the lemma.

Because of the strict preference for  $T$ , the probabilities that  $u$  and  $v$  are assigned 1 for the task  $T'$  are

$$\Pr[\mathcal{S}(u)_{T'} = 1] = (1 - p_u)p'_u$$

$$\Pr[\mathcal{S}(v)_{T'} = 1] = (1 - p_v)p'_v$$

The difference between them is

$$\begin{aligned} \Pr[\mathcal{S}(u)_{T'} = 1] - \Pr[\mathcal{S}(v)_{T'} = 1] &= (1 - p_u)p'_u - (1 - p_v)p'_v \\ &= p'_u - p_u p'_u - p'_v + p_v p'_v \\ &= p'_u - p'_v + p_v p'_v - p_u p'_u \end{aligned}$$

Notice that if  $\mathcal{D}'(u, v) = 0$ , which implies that  $p'_u = p'_v$ , and  $p_u \neq p_v$ , then this quantity is non-zero, giving the desired contradiction for all fair  $C'$  and any  $C$  that assigns  $p_u \neq p_v$ , which can be constructed per Corollary 18.

However, if  $\mathcal{D}'(u, v) \neq 0$ , take  $C'$  such that  $|p'_u - p'_v| = \mathcal{D}'(u, v)$  and denote the distance  $|p'_u - p'_v| = m'$ , and without loss of generality, assume that  $p'_u > p'_v$  and  $p_u < p_v$ ,

$$\Pr[\mathcal{S}(u)_{T'} = 1] - \Pr[\mathcal{S}(v)_{T'} = 1] = m' + p_v p'_v - p_u p'_u$$

Then to violate fairness for  $T'$ , it suffices to show that  $p_v p'_v > p_u p'_u$ . Write  $p_v = \alpha p_u$  where  $\alpha > 1$ ,

$$\alpha p_u p'_v > p_u p'_u$$

<sup>7</sup> A metric  $\mathcal{D}$  is said to be *non-trivial* if there exists at least one pair,  $u, v \in U$  such that  $\mathcal{D}(u, v) \notin \{0, 1\}$ .

$$\alpha p'_v > p'_u$$

Thus it is sufficient to show that we can choose  $p_u, p_v$  such that  $\alpha > \frac{p'_u}{p'_v}$ . Constrained only by the requirements that  $p_u < p_v$  and  $|p_u - p_v| \leq \mathcal{D}(u, v)$ , we may choose  $p_u, p_v$  to obtain an arbitrarily large  $\alpha = \frac{p_v}{p_u}$  by Corollary 19. Thus there exist a pair of fair classifiers  $C, C'$  which when combined with strictly ordered task-competitive composition violate multiple task fairness. ◀

► **Theorem 8.** *For any two tasks  $T$  and  $T'$  with nontrivial metrics  $\mathcal{D}$  and  $\mathcal{D}'$  respectively, there exists a set  $\mathcal{C}$  of classifiers which are individually fair in isolation but when combined with task-competitive composition violate multiple task fairness for any tie-breaking function.*

**Proof.** Consider a pair of classifiers  $C, C'$  for the two tasks. Let  $p_u$  denote the probability that  $C$  assigns 1 to  $u$ , and analogously let  $p_v, p'_u, p'_v$  denote this quantity for the other classifier and element combinations. As noted before, for convenience of notation, write  $\mathbb{B}_u(T)$  to indicate the preference for each (element, outcome) pair, that is the probability that given the choice between  $T$  or the alternative outcome  $T'$ ,  $T$  is chosen. Note that in this system, for each element  $\mathbb{B}_u(T) + \mathbb{B}_u(T') = 1$ .

Note that if  $\mathbb{B}_w(T) = 1 \forall w \in U$  or  $\mathbb{B}_w(T') = 1 \forall w \in U$ , the setting is exactly as described in Lemma 7. Thus we need only argue for the two following cases:

1. Case  $\mathbb{B}_u(T) = \mathbb{B}_v(T) \neq 1$ . We can write an expression for the probability that each element is assigned to task  $T$ :

$$\Pr[\mathcal{S}(u)_T = 1] = p_u(1 - p'_u) + p_u p'_u \mathbb{B}_u(T)$$

$$\Pr[\mathcal{S}(v)_T = 1] = p_v(1 - p'_v) + p_v p'_v \mathbb{B}_v(T)$$

So the difference in probabilities is

$$\begin{aligned} \Pr[\mathcal{S}(u)_T = 1] - \Pr[\mathcal{S}(v)_T = 1] &= p_u(1 - p'_u) + p_u p'_u \mathbb{B}_u(T) - p_v(1 - p'_v) - p_v p'_v \mathbb{B}_v(T) \\ &= p_u - p_v + p_v p'_v - p_u p'_u + p_u p'_u \mathbb{B}_u(T) - p_v p'_v \mathbb{B}_v(T) \\ &= p_u - p_v + (p_v p'_v - p_u p'_u)(1 - \mathbb{B}_u(T)) \end{aligned}$$

By our assumption that  $\mathbb{B}_u(T) \neq 1$ , we proceed analogously to the proof of Lemma 7 choosing  $C'$  such that  $p_v p'_v > p_u p'_u$  and choosing  $C$  to ensure that  $p_u - p_v = \mathcal{D}(u, v)$  to achieve unfairness for  $T$ .

2. Case  $\mathbb{B}_u(T) \neq \mathbb{B}_v(T)$ . Assume without loss of generality that  $\mathbb{B}_u(T) \neq 1$ . Recall the difference in probability of assignment of 1 for the first task in terms of  $\mathbb{B}$ :

$$= p_u - p_v + p_v p'_v (1 - \mathbb{B}_v(T)) - p_u p'_u (1 - \mathbb{B}_u(T))$$

Choose  $C$  such that  $p_u - p_v = \mathcal{D}(u, v)$  (or if there is no such individually fair  $C$ , choose the individually fair  $C$  which maximizes the distance between  $u$  and  $v$ ). So it suffices to show that we can select  $C'$  such that  $p_v p'_v (1 - \mathbb{B}_v(T)) - p_u p'_u (1 - \mathbb{B}_u(T)) > 0$ . As before, write  $p_u = \alpha p_v$  where  $\alpha > 1$ . We require:

$$p_v p'_v (1 - \mathbb{B}_v(T)) > \alpha p_v p'_u (1 - \mathbb{B}_u(T))$$

$$p'_v (1 - \mathbb{B}_v(T)) > \alpha p'_u (1 - \mathbb{B}_u(T))$$

Writing  $\beta = (1 - \mathbb{B}_v(T))/(1 - \mathbb{B}_u(T))$  (recall that  $\mathbb{B}_u(T) \neq 1$  so there is no division by zero), we require

$$p'_v \beta > \alpha p'_u$$

$$\beta/\alpha > p'_u/p'_v$$

Constrained only by  $|p'_u - p'_v| \leq \mathcal{D}'(u, v)$ , we can choose  $p'_u, p'_v$  to be any arbitrary positive ratio per Corollary 19, thus we can select a satisfactory  $C'$  to exceed the allowed distance.

Thus we have shown that for the cases where the tie-breaking functions are identical for  $u$  and  $v$  and when the tie-breaking functions are different, there always exists a pair of classifiers  $C, C'$  which are fair in isolation, but when combined in task-competitive composition do not satisfy multiple task fairness which completes the proof. ◀

The intuition for unfairness in such a strictly ordered composition is that each task inflicts its preferences on subsequent tasks, and this intuition extends to more complicated tie-breaking functions and individuals with positive distances in both tasks. Our intuition suggests that the situation in Theorem 8 is not contrived and occurs often in practice, and moreover that small relaxations will not be sufficient to alleviate this problem, as the phenomenon has been observed empirically [3, 17, 15]. We include a small simulated example in the Appendix of the full version to illustrate the potential magnitude and frequency of such fairness violations.

### 3.2 Simple Fair Multiple-task Composition

Fortunately, there is a general purpose mechanism for the single slot composition problem which requires no additional information in learning each classifier and no additional coordination between the classifiers.<sup>8</sup> The rough procedure for `RandomizeThenClassify` (Algorithm 1) is to fix a fair classifier for each task, fix a probability distribution over the tasks, sample a task from the distribution, and then run the fair classifier for that task. `RandomizeThenClassify` has several nice properties: it requires no coordination in the training of the classifiers, it preserves the ordering and relative distance of elements by each classifier, and it can be implemented by a platform or other third party, rather than requiring the explicit cooperation of all classifiers. The primary downside of `RandomizeThenClassify` is that it reduces allocation (the total number of positive classifications) for classifiers trained with the expectation of being run independently.

## 4 Functional Composition

In *Functional Composition*, the outputs of multiple classifiers are combined through logical operations to produce a single output for a single task. A significant consideration in functional composition is determining which outcomes are relevant for fairness and at which point(s) fairness should be measured. For example, (possibly different) classifiers for admitting students to different colleges are composed to determine whether the student is accepted to at least one college. In this case, the function is “OR”, the classifiers are for the same task, and hence conform to the same metric, and this is the same metric one might use for defining

<sup>8</sup> See section Appendix Section 6.4 in the full version for another mechanism which requires coordination between the classifiers.

fairness of the system as a whole. Alternatively, the system may compose the classifier for admission with the classifier for determining financial aid. In this case the function is “AND”, the classifiers are for different tasks, with different metrics, and we may use scholastic ability or some other appropriate output metric for evaluating overall fairness of the system.

#### 4.1 Same-task Functional Composition

In this section, we consider the motivating example of college admissions. When secondary school students apply for college admission, they usually apply to more than one institution to increase their odds of admission to at least one college. Consider a universe of students  $U$  applying to college in a particular year, each with intrinsic qualification  $q_u \in [0, 1]$ ,  $\forall u \in U$ . We define  $\mathcal{D}(u, v) = |q_u - q_v| \forall u, v \in U$ .  $\mathcal{C}$  is the set of colleges and assume each college  $C_i \in \mathcal{C}$  admits students fairly with respect to  $\mathcal{D}$ . The system of schools is considered OR-fair if the indicator variable  $x_u$ , which indicates whether or not student  $u$  is admitted to at least one school, satisfies individual fairness under this same metric. More formally,

► **Definition 9 (OR Fairness).** Given a (universe, task) pair with metric  $\mathcal{D}$ , and a set of classifiers  $\mathcal{C}$  we define the indicator

$$x_u = \begin{cases} 1 & \text{if } \sum_{C_i \in \mathcal{C}} C_i(x) \geq 1 \\ 0 & \text{otherwise} \end{cases}$$

which indicates whether at least one positive classification occurred. Define  $\tilde{x}_u = \Pr[x_u = 1] = 1 - \prod_{C_i \in \mathcal{C}} (1 - \Pr[C_i(u) = 1])$ . Then the composition of the set of classifiers  $\mathcal{C}$  satisfies *OR Fairness* if  $\mathcal{D}(u, v) \geq d(\tilde{x}_u, \tilde{x}_v)$  for all  $u, v \in U$ .

The OR Fairness setting matches well to tasks where individuals primarily *benefit* from one positive classification for a particular task.<sup>9</sup> As mentioned above, examples of such tasks include gaining access to credit or a home loan, admission to university, access to qualified legal representation, access to employment, *etc.*<sup>10</sup> Although in some cases more than one acceptance may have positive impact, for example a person with more than one job offer may use the second offer to negotiate a better salary, the core problem is (arguably) whether or not at least one job is acquired.

Returning to the example of college admissions, even with the strong assumption that each college fairly evaluates its applicants, there are still several potential sources of unfairness in the resulting system. In particular, if students apply to different numbers of colleges or colleges with different admission rates, we would expect that their probabilities of acceptance to at least one college will be different. The more subtle scenario from the perspective of composition is when students apply to the *same* set of colleges.

Even in this restricted setting, it is still possible for a set of classifiers for the same task to violate OR fairness. The key observation is that for elements with positive distance, the difference in their expectation of acceptance by at least one classifier does not diverge linearly in the number of classifiers included in the composition. As the number of classifiers increases, the probabilities of positive classification by at least one classifier for any pair eventually converge. However, in practice, we expect students to apply to perhaps five or 10 colleges, so it is desirable to characterize when small systems are robust to such composition.

<sup>9</sup> We may conversely define NOR Fairness to take  $\neg x_u$ , and this setting more naturally corresponds to cases where not being classified as positive is desirable.

<sup>10</sup> [1] considers what boils down to AND-fairness for Equal Opportunity and presents an excellent collection of evocative example scenarios.

► **Theorem 10.** *For any (universe, task) pair with a non-trivial metric  $\mathcal{D}$ , there exists a set of individually fair classifiers  $\mathcal{C}$  which do not satisfy OR Fairness, even if each element in  $U$  is classified by all  $C_i \in \mathcal{C}$ .*

The proof of Theorem 10 follows from a straightforward analysis of the difference in probability of at least one positive classification.<sup>11</sup> The good news is that there exist non-trivial conditions for sets of small numbers of classifiers where OR Fairness is satisfied:

► **Lemma 11.** *Fix a set  $\mathcal{C}$  of fair classifiers, and let  $x_w$  for  $w \in U$  be the indicator variable as in Definition 9. If  $\mathbb{E}[x_w] \geq 1/2$  for all  $w \in U$ , then the set of classifiers  $\mathcal{C} \cup \{C'\}$  satisfies OR fairness if  $C'$  satisfies individual fairness under the same metric and  $\Pr[C'(w) = 1] \geq \frac{1}{2}$  for all  $w \in U$ .*

This lemma is useful for determining that a system is free from same-task divergence, as it is possible to reason about an “OR of ORs”, and more generally an “OR” of any fair components of sufficient weight.

Functional composition can also be used to reason about settings where classification procedures for different tasks are used to determine the outcome for a single task. For example, in order to attend a particular college, a student must be admitted *and* receive sufficient financial aid to afford tuition and living expenses. Financial need and academic qualification clearly have different metrics, and in such settings, a significant challenge is to understand how the input metrics relate to the relevant output metric. Without careful reasoning about the interaction between these tasks, it is very easy to end up with systems which violate individual fairness, even if they are constructed from individually fair components. (See Section 4.2 in the full version for more details.)

## 5 Dependent Composition

Thus far, we have restricted our attention to the mode of operation in which classifiers act on the entire universe of individuals at once and each individual’s outcome is decided independently. In practice, however, this is an unlikely scenario, as classifiers may be acting as a selection mechanism for a fixed number of elements, may operate on elements in arbitrary order, or may operate on only a subset of the universe. In this section, we consider the case in which the classification outcomes received by individuals are not independent. Slightly abusing the term “composition,” these problems can be viewed as a composition of the classifications of elements of the universe. We roughly divide these topics into Cohort Selection problems, when a set of exactly  $n$  individuals must be selected from the universe, and Universe Subset problems, when only a subset of the relevant universe for the task is under the influence of the classifier we wish to analyze or construct. Within these two problems we consider several relevant settings:

**Online versus offline:** Advertising decisions for online ads must be made immediately upon impression and employers must render employment decisions quickly or risk losing out on potential employees or taking too long to fill a position.

**Random versus adversarial ordering:** The order in which individuals apply for an open job may be influenced by their social connections with existing employees, which impacts how quickly they hear about the job opening.

<sup>11</sup> See Appendix Section 4 in the full version for the complete proof.

**Known versus unknown subset or universe size:** An advertiser may know the average number of interested individuals who visit a website on a particular day, but be uncertain on any particular day of the exact number.

**Constrained versus unconstrained selection:** In many settings there are arbitrary constraints placed on selection of individuals for a task which are unrelated to the qualification or metric for that task. For example, to cover operating costs, a college may need at least  $n/2$  of the  $n$  students in a class to be able to pay full tuition.

In dependent composition problems, it is important, when computing distances between distributions over outcomes, to pay careful attention to the source of randomness. Taking inspiration from the experiment setup found in many cryptographic definitions, we formally define two problems, Universe Subset Classification and Cohort Selection, (included in Definitions 13 and 14 in the Appendix). In particular, it is important to understand the randomness used to decide an ordering or a subset, as once an ordering or subset is fixed, reasoning about fairness is impossible, as a particular individual may be arbitrarily included or excluded.

## 5.1 Basic Offline Cohort Selection

First we consider the simplest version of the cohort selection problem: choosing a cohort of  $n$  individuals from the universe  $U$  when the entire universe is known and decisions are made offline. A simple solution is to choose a permutation of the elements in  $U$  uniformly at random, and then apply a fair classifier  $C$  until  $n$  are selected or selecting the last few elements from the end of the list if  $n$  have not yet been selected. With some careful bookkeeping, we show that this mechanism is individually fair for any individually fair input classifier. (See Algorithms 2 and 3 in the Appendix below; a complete analysis is included in Appendix Section 6 in the full version.)

## 5.2 More complicated settings

In this extended abstract, we omit a full discussion of the more complicated dependent composition scenarios, but briefly summarize several settings to build intuition.

► **Theorem 12.** *If the ordering of the stream is adversarial, but  $|U|$  is unknown, then there exists no solution to the online cohort selection problem.*

The intuition for the proof follows from imagining that a fair classification process exists for an ordering of size  $n$  and realizing that this precludes fair classification of a list of size  $n + 1$ , as the classification procedure cannot distinguish between the two cases.

### Constrained cohort selection

Next we consider the problem of selecting a cohort with an external requirement that some fraction of the selected set is from a particular subgroup. That is, given a universe  $U$ , and  $p \in [0, 1]$ , and a subset  $A \subset U$ , select a cohort of  $n$  elements such that at least a  $p$  fraction of the elements selected are in  $A$ . This problem captures situations in which external requirements cannot be ignored. For example, if a certain budget must be met, and only some members of the universe contribute to the budget, or if legally a certain fraction of people selected must meet some criterion (as in, demographic parity). In the full version, we characterize a broad range of settings where the constrained cohort selection problem cannot be solved fairly.

To build intuition, suppose the universe  $U$  is partitioned into sets  $A$  and  $B$ , where  $n/2 = |A| = |B|/5$ . Suppose further that the populations have the same distribution on ability, so that the set  $B$  is a “blown up” version of  $A$ , meaning that for each element  $u \in A$  there are 5 corresponding elements  $V_u = \{v_{u,1}, \dots, v_{u,5}\}$  such that  $\mathcal{D}(u, v_{u,i}) = 0$ ,  $1 \leq i \leq 5$ ,  $\forall u, u' \in A V_u \cap V_{u'} = \emptyset$ , and  $B = \cup_{u \in A} V_u$ . Let  $p = \frac{1}{2}$ . The constraint requires all of  $A$  to be selected; that is, each element of  $A$  has probability 1 of selection. In contrast, the average probability of selection for an element of  $B$  is  $\frac{1}{5}$ . Therefore, there exists  $v \in B$  with selection probability at most  $1/5$ . Letting  $u \in A$  such that  $v \in V_u$ , we have  $\mathcal{D}(u, v) = 0$  but the difference in probability of selection is at least  $\frac{4}{5}$ . We give a more complete characterization of the problem and impossibilities in the full version in Appendix Section 6.3 .

## 6 Extensions to Group Fairness

In general, the results discussed above for composition of individual fairness extend to group fairness definitions; however, there are several issues and technicalities unique to group fairness definitions which we now discuss.

### Technicalities

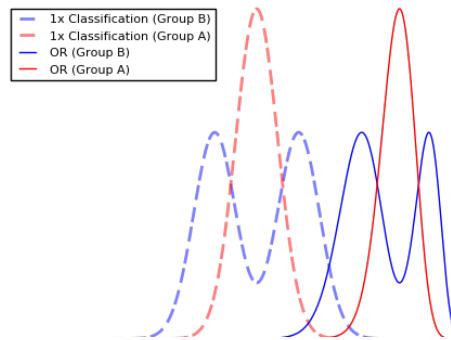
Consider the following simple universe: for a particular  $z \in \mathcal{Z}$ , group  $B$  is unimodal, having only elements with medium qualification  $q_m$ , while group  $A$  is bimodal, with half of its elements having low qualification  $q_l$  and half having high qualification  $q_h$ . Choosing  $p_h = 1$ ,  $p_m = .75$ , and  $p_l = .5$  satisfies Conditional Parity for a single application. However, for the OR of two applications, the squares diverge ( $.9375 \neq .875$ ), violating conditional parity (see Figure 1). Note, however, that all of the individuals with  $\mathbf{z} = z$  have been drawn closer together under composition, and none have been pulled further apart. This simple observation implies that in some cases we may observe failures under composition for conditional parity, even when individual fairness is satisfied. In order to satisfy Conditional Parity under OR-composition, the classifier could sacrifice accuracy by treating all individuals with  $\mathbf{z} = z$  equally. However, this necessarily discards useful information about the individuals in  $A$  to satisfy a technicality.

### Subgroup Subtleties

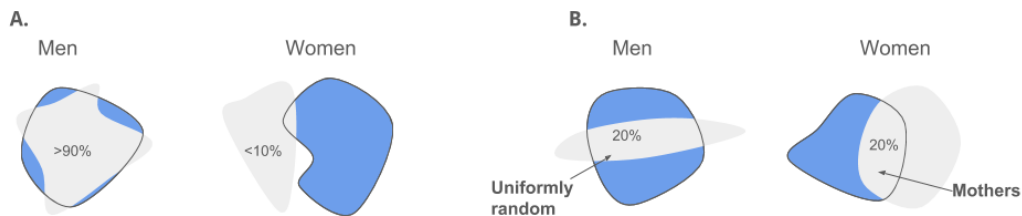
There are many cases where failing to satisfy conditional parity under task-competitive composition is clearly a violation of our intuitive notion of group fairness. However, conditional parity is not always a reliable test for fairness at the subgroup level under composition. In general, we expect conditional parity based definitions of group fairness to detect unfairness in multiple task compositions reasonably well when there is an obvious interaction between protected groups and task qualification, as observed empirically in [17] and [3]. For example, let’s return to our advertising example where home-goods advertisers have no protected set, but high-paying jobs have gender as a protected attribute. Under composition, home-goods out-bidding high-paying jobs ads for women will clearly violate the conditional parity condition for the job ads (see Figure 2).

However, suppose that, in response to gender disparity caused by task-competitive composition, classifiers iteratively adjust their bids to try to achieve Conditional Parity. This may cause them to *learn themselves* into a state that satisfies Conditional Parity with respect to gender, but behaves poorly for a socially meaningful subgroup (see Figure 3.) For example, if home goods advertisers aggressively advertise to women who are new parents





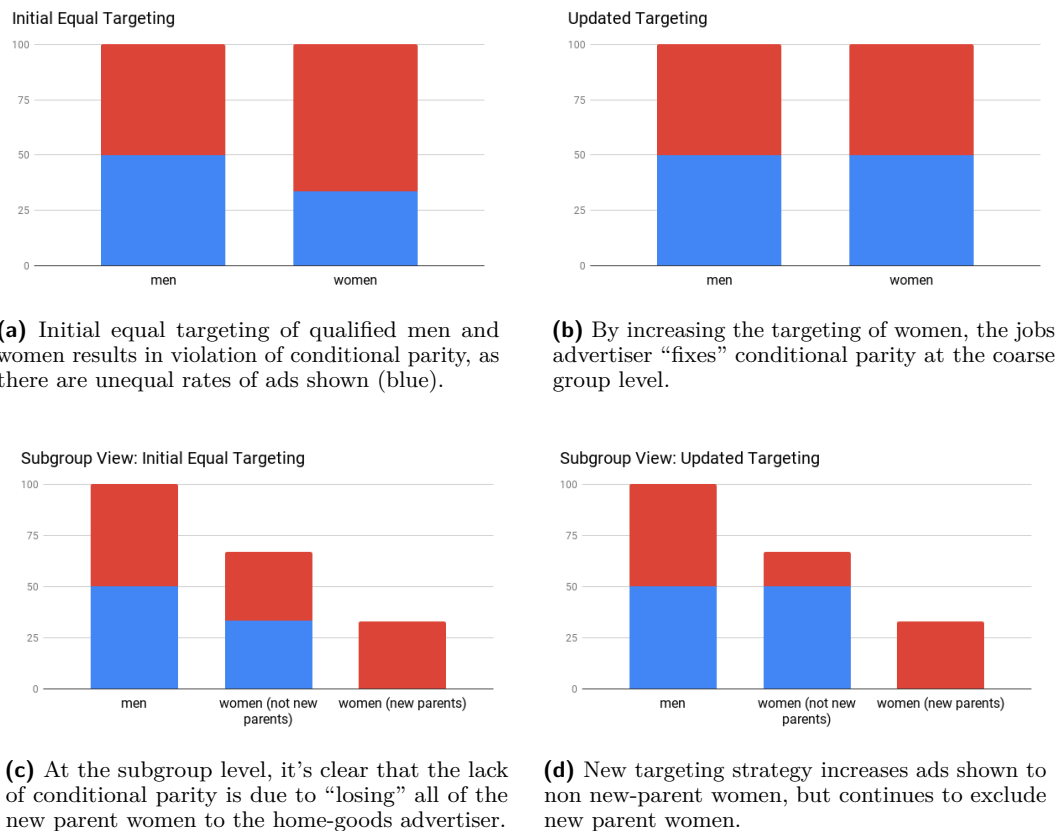
■ **Figure 1** An illustration of the shift in groups from a single classification to the OR of two applications of the same classifier. Although the two groups originally had the same mean probability of positive classification, this breaks down under OR composition.



■ **Figure 2 A.** When the two tasks are related, one will ‘claim’ a larger fraction of one gender than another, leading to a smaller fraction of men remaining for classification in the other task (shown in blue). Conditional parity will detect this unfairness. **B.** When the tasks are unrelated, one task may ‘claim’ the same fraction of people in each group, but potentially select a socially meaningful subgroup, eg parents. Conditional parity will fail to detect this subgroup unfairness, unless subgroups, including any subgroups targeted by classifiers composed with, are explicitly accounted for.

(because their life-time value ( $\mathcal{Z}$ ) to the advertiser is the highest of all universe elements), then a competing advertiser for jobs, noticing that its usual strategy of recruiting all people with skill level  $\mathbf{z}' = z'$  equally is failing to reach enough women, bids more aggressively on women. By bidding more aggressively, the advertiser increases the probability of showing ads to women (for example by outbidding low-value competition), but not to women who are bid for by the home goods advertiser (a high-value competitor), resulting in a high concentration of ads for women who are *not* mothers, while still failing to reach women who *are* mothers. Furthermore, the systematic exclusion of mothers from job advertisements can, over time, be even more problematic, as it may contribute to the stalling of careers. In this case, the system discriminates against mothers without necessarily discriminating against fathers.

Although problematic (large) subgroup semantics are part of the motivation for [11, 7] and exclusion of subgroups is not only a composition problem, the added danger in composition is that the features describing this subset may be missing from the feature set of the jobs classifier, rendering the protections proposed in [11] and [7] ineffective. In particular, we expect that sensitive attributes like parental status are unlikely to appear (or are illegal to collect) in employment-related training or testing datasets, but may be legitimately targeted by other competing advertisers.



■ **Figure 3** Home-goods advertisers aggressively target mothers, out-bidding the jobs advertiser. When the jobs advertiser bids more aggressively on “women” (b) the overall rate of ads shown to “women” increases, but mothers may still be excluded (d), so  $\Pr[\text{ad} \mid \text{qualified, woman}] > \Pr[\text{ad} \mid \text{qualified, mother}]$ .

## References

- 1 Amanda Bower, Sarah N. Kitchen, Laura Niss, Martin J. Strauss, Alexander Vargas, and Suresh Venkatasubramanian. Fair Pipelines. *CoRR*, abs/1707.00391, 2017. [arXiv:1707.00391](#).
- 2 Alexandra Chouldechova. Fair prediction with disparate impact: A study of bias in recidivism prediction instruments. *arXiv preprint*, 2017. [arXiv:1703.00056](#).
- 3 Amit Datta, Michael Carl Tschantz, and Anupam Datta. Automated experiments on ad privacy settings. *Proceedings on Privacy Enhancing Technologies*, 2015(1):92–112, 2015.
- 4 Cynthia Dwork, Moritz Hardt, Toniann Pitassi, Omer Reingold, and Richard Zemel. Fairness through awareness. In *Proceedings of the 3rd innovations in theoretical computer science conference*, pages 214–226. ACM, 2012.
- 5 Stephen Gillen, Christopher Jung, Michael Kearns, and Aaron Roth. Online Learning with an Unknown Fairness Metric. *arXiv preprint*, 2018. [arXiv:1802.06936](#).
- 6 Moritz Hardt, Eric Price, Nati Srebro, et al. Equality of opportunity in supervised learning. In *Advances in Neural Information Processing Systems*, pages 3315–3323, 2016.
- 7 Ursula Hébert-Johnson, Michael P Kim, Omer Reingold, and Guy N Rothblum. Calibration for the (Computationally-Identifiable) Masses. *arXiv preprint*, 2017. [arXiv:1711.08513](#).

- 8 Lily Hu and Yiling Chen. Fairness at Equilibrium in the Labor Market. *CoRR*, abs/1707.01590, 2017. [arXiv:1707.01590](#).
- 9 Faisal Kamiran and Toon Calders. Classifying without discriminating. In *Computer, Control and Communication, 2009. IC4 2009. 2nd International Conference on*, pages 1–6. IEEE, 2009.
- 10 Toshihiro Kamishima, Shotaro Akaho, and Jun Sakuma. Fairness-aware learning through regularization approach. In *Data Mining Workshops (ICDMW), 2011 IEEE 11th International Conference on*, pages 643–650. IEEE, 2011.
- 11 Michael Kearns, Seth Neel, Aaron Roth, and Zhiwei Steven Wu. Preventing fairness gerrymandering: Auditing and learning for subgroup fairness. *arXiv preprint*, 2017. [arXiv:1711.05144](#).
- 12 Niki Kilbertus, Mateo Rojas-Carulla, Giambattista Parascandolo, Moritz Hardt, Dominik Janzing, and Bernhard Schölkopf. Avoiding Discrimination through Causal Reasoning. *arXiv preprint*, 2017. [arXiv:1706.02744](#).
- 13 Michael P Kim, Omer Reingold, and Guy N Rothblum. Fairness Through Computationally-Bounded Awareness. *arXiv preprint*, 2018. [arXiv:1803.03239](#).
- 14 Jon M. Kleinberg, Sendhil Mullainathan, and Manish Raghavan. Inherent Trade-Offs in the Fair Determination of Risk Scores. *CoRR*, abs/1609.05807, 2016. [arXiv:1609.05807](#).
- 15 Peter Kuhn and Kailing Shen. Gender discrimination in job ads: Evidence from china. *The Quarterly Journal of Economics*, 128(1):287–336, 2012.
- 16 Matt J Kusner, Joshua R Loftus, Chris Russell, and Ricardo Silva. Counterfactual Fairness. *arXiv preprint*, 2017. [arXiv:1703.06856](#).
- 17 Anja Lambrecht and Catherine E Tucker. Algorithmic Bias? An Empirical Study into Apparent Gender-Based Discrimination in the Display of STEM Career Ads, 2016.
- 18 Lydia T Liu, Sarah Dean, Esther Rolf, Max Simchowitz, and Moritz Hardt. Delayed Impact of Fair Machine Learning. *arXiv preprint*, 2018. [arXiv:1803.04383](#).
- 19 David Madras, Elliot Creager, Toniann Pitassi, and Richard Zemel. Learning Adversarially Fair and Transferable Representations. *arXiv preprint*, 2018. [arXiv:1802.06309](#).
- 20 Dino Pedreshi, Salvatore Ruggieri, and Franco Turini. Discrimination-aware data mining. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 560–568. ACM, 2008.
- 21 Ya’acov Ritov, Yuekai Sun, and Ruofei Zhao. On conditional parity as a notion of non-discrimination in machine learning. *arXiv preprint*, 2017. [arXiv:1706.08519](#).
- 22 Rich Zemel, Yu Wu, Kevin Swersky, Toni Pitassi, and Cynthia Dwork. Learning fair representations. In *Proceedings of the 30th International Conference on Machine Learning (ICML-13)*, pages 325–333, 2013.

## **A** Appendix

### **A.1** Algorithm for Task-Competitive Composition

RandomizeThenClassify, Algorithm 1 has several nice properties. First, it requires no coordination in the training of the classifiers. In particular, it does not require any sharing of objective functions. Second, it preserves the ordering of elements by each classifier. That is, if  $\Pr[C_i(u) = 1] > \Pr[C_i(v) = 1]$  then  $\Pr[\text{RandomizeThenClassify}(u)_i = 1] > \Pr[\text{RandomizeThenClassify}(v)_i = 1]$ . Finally, it can be implemented by a platform or other third party, rather than requiring the explicit cooperation of all classifiers. The primary downside of RandomizeThenClassify is that it drastically reduces allocation (the total number of positive classifications) for classifiers trained with the expectation of being run independently.

---

**Algorithm 1** RandomizeThenClassify.

---

**Input:** universe element  $u \in U$ , set of fair classifiers  $\mathcal{C}$  (possibly for distinct tasks) operating on  $U$ , probability distribution over tasks  $\mathcal{X} \in \Delta(\mathcal{C})$   
 $x \leftarrow 0^{|\mathcal{C}|}$   
 $C_t \sim \mathcal{X}$   
**if**  $C_t(u) = 1$  **then**  
     $x_t = 1$   
**end if**  
return  $x$

---



---

**Algorithm 2** PermuteThenClassify.

---

**Input:**  $n \leftarrow$  the number of elements to select  
 $C \leftarrow$  a classifier  $C : U \times \{0, 1\}^* \rightarrow \{0, 1\}$   
 $\pi \sim S_{|U|}$  a random permutation from the symmetric group on  $|U|$   
 $L \leftarrow \pi(U)$  An ordered set of elements  
 $M \leftarrow \emptyset$   
**while**  $|M| < n$ : **do**  
     $u \leftarrow \text{pop}(L)$   
    **if**  $C(u) = 1$  **then**  
         $M \leftarrow M \cup \{u\}$   
    **end if**  
    **if**  $n - |M| \geq |L|$  **then**  
        *// the end condition*  
         $M \leftarrow M \cup \{u\}$   
    **end if**  
**end while**  
return  $M$

---

## A.2 Algorithms for Cohort Selection

PermuteThenClassify, Algorithm 2, works through a list initialized to a random permutation  $\pi(U)$ , classifying elements one at a time and independently until either (1)  $n$  elements have been selected or (2) the number of remaining elements in the list equals the number of remaining spots to be filled. Case (2) is referred to as the “end condition”. Elements in the “end condition” are selected with probability 1.

WeightedSampling, Algorithm 3, chooses sets of elements with probability proportional to their weight under a fair classifier. This prevents the arbitrary behavior of the end condition in case the classifier is poorly tuned for the specific number of desired elements.

## A.3 Universe Subset Problems

► **Definition 13** (Universe Subset Classification Problem). Given a universe  $U$ , let  $\mathcal{Y}$  be a distribution over subsets of  $U$ . Let  $\mathcal{X} = \{\mathcal{X}(V)\}_{V \subseteq U}$  be a family of distributions, one for each subset of  $U$ , where  $\mathcal{X}(V)$  is a distribution on permutations of the elements of  $V$ . Let  $\Pi(2^U)$  denote the set of permutations on subsets of  $U$ . Formally, for a system  $\mathcal{S} : \Pi(2^U) \times \{0, 1\}^* \rightarrow U^*$ , we define **Experiment**( $\mathcal{S}, \mathcal{X}, \mathcal{Y}, u$ ) as follows:

1. Choose  $r \sim \{0, 1\}^*$
2. Choose  $V \sim \mathcal{Y}$

**Algorithm 3** WeightedSampling.

---

**Input:**  $n \leftarrow$  the number of elements to select  
 $C \leftarrow$  a classifier  $C : U \times r \rightarrow \{0, 1\}$   
 $L \leftarrow$  the set of all subsets of  $U$  of size  $n$   
**for**  $l \in L$  **do**  
     $w(l) \leftarrow \sum_{u \in l} \mathbb{E}[C(u)]$  // set the weight of each set  
    Define  $\mathcal{X} \in \Delta(L)$  such that  $\forall l \in L$ , the weight of  $l$  under  $\mathcal{X}$  is  $\frac{w(l)}{\sum_{l' \in L} w(l')}$   
     $M \sim \mathcal{X}$  // Sample a set of size  $n$  according to  $\mathcal{X}$   
**end for**  
return  $M$

---

3. Choose  $\pi \sim \mathcal{X}(V)$
4. Run  $\mathcal{S}$  on  $\pi$  with randomness  $r$ , and output 1 if  $u$  is selected (positively classified).

The system  $\mathcal{S}$  is individually fair and a solution to the Universe Subset Classification Problem for a particular  $(\mathcal{X}, \mathcal{Y})$  pair if for all  $u, v \in U$ ,

$$|\mathbb{E}[\text{Experiment}(\mathcal{S}, \mathcal{X}, \mathcal{Y}, u)] - \mathbb{E}[\text{Experiment}(\mathcal{S}, \mathcal{X}, \mathcal{Y}, v)]| \leq \mathcal{D}(u, v)$$

Note that for any distinct individuals  $u, v \in U$ , in any given run of the experiment  $V$  may contain  $u, v$ , neither or both.

► **Definition 14** (Cohort Selection Problem). The Cohort Selection Problem is identical to the Universe Subset Classification Problem, except the system is limited to choosing exactly  $n$  individuals.

► **Lemma 15.** *Given an instance of the universe subset classification problem (Definition 13) where  $\mathcal{Y}$  assigns positive weight to all elements  $w \in U$ , the following procedure applied to any individually fair classifier  $C$  which solely controls outcomes for a particular task will result in fair classification under the input distribution  $\mathcal{Y}$ .*

**Procedure:** for each  $w \in U$ , let  $q_w$  denote the probability that  $w$  appears in  $V$ . Let  $q_{\min} = \min_w q_w$ . For each element  $w \in V$ , with probability  $q_{\min}/q_w$  classify  $w$  normally, otherwise output the default for no classification.

**Proof.** Let  $u = \operatorname{argmin}_w(q_w)$ . Then  $u$  will be classified positively with probability  $p_u q_{\min}$  where probability is taken over  $\mathcal{Y}$  and  $C$ . All other elements  $v \in V$  will be classified positively with probability  $q_v(q_{\min}/q_v)p_v = p_v q_{\min}$ . As positive classification by  $C$  is the only way to get a positive outcome for the task, reasoning about  $|p_v - p_u|$  is sufficient to ensure fairness. Therefore, if  $|p_v - p_u| \leq \mathcal{D}(u, v)$ , then the distance under this procedure is also  $\leq \mathcal{D}(u, v)$ . ◀

## A.4 Construction of Fair Classifiers

► **Lemma 16.** *Let  $V$  be a (possibly empty) subset of  $U$ . If there exists a classifier  $C : V \times \{0, 1\}^* \rightarrow \{0, 1\}$  such that  $\mathcal{D}(u, v) \geq d(\tilde{C}(u), \tilde{C}(v))$  for all  $u, v \in V$ , then for any  $x \in U \setminus V$  there exists classifier  $C' : V \cup \{x\} \times \{0, 1\}^* \rightarrow \{0, 1\}$  such that  $\mathcal{D}(u, v) \geq d(\tilde{C}(u), \tilde{C}(v))$  for all  $u, v \in U$ , which has identical behavior to  $C$  on  $V$ .*

**Proof.** For  $V = \emptyset$ , any value  $p_x$  suffices to fairly classify  $x$ . For  $|V| = 1$ , choosing any  $p_x$  such that  $|p_v - p_x| \leq \mathcal{D}(v, x)$  for  $v \in V$  suffices.

---

**Algorithm 4** FairAddition( $\mathcal{D}, V, p_t, C, x$ ).

---

**Input:** metric  $\mathcal{D}$  for universe  $U$ , a subset  $V \subset U$ , target probability  $p_t$ , an individually fair classifier  $C : V \times \{0, 1\}^* \rightarrow \{0, 1\}$ , a target element  $x \in U \setminus V$  to be added to  $C$ .

Initialize  $L \leftarrow V$

$\hat{p}_x \leftarrow p_t$

**for**  $l \in L$  **do**

$dist \leftarrow \mathcal{D}(l, x)$

**if**  $dist < p_l - \hat{p}_x$  **then**

$\hat{p}_x \leftarrow p_l - dist$

**else if**  $dist < \hat{p}_x - p_l$  **then**

$\hat{p}_x \leftarrow p_l + dist$

**end if**

**end for**

**return**  $\hat{p}_x$

---

For  $|V| \geq 2$ , apply the procedure outlined in Algorithm 4 taking  $p_t$  to be the probability of positive classification of  $x$ 's nearest neighbor in  $V$  under  $C$ . As usual, we take  $p_w$  to be the probability that  $C$  positively classifies element  $w$ .

Notice that Algorithm 4 only modifies  $\hat{p}_x$ , and that  $\hat{p}_x$  is only changed if a distance constraint is violated. Thus it is sufficient to confirm that on each modification to  $\hat{p}_x$ , no distance constraints between  $x$  and elements in the opposite direction of the move are violated.

Without loss of generality, assume that  $\hat{p}_x$  is decreased to move within an acceptable distance of  $u$ , that is  $\hat{p}_x \geq p_u$ . It is sufficient to show that for all  $v$  such that  $p_v > \hat{p}_x$  that no distances are violated. Consider any such  $v$ . By construction  $\hat{p}_x - p_u = \mathcal{D}(u, x)$ , and  $p_v - p_u \leq \mathcal{D}(u, v)$ . From triangle inequality, we also have that  $\mathcal{D}(u, v) \leq \mathcal{D}(u, x) + \mathcal{D}(x, v)$ . Substituting, and using that  $p_v \geq \hat{p}_x \geq p_u$ :

$$\mathcal{D}(u, v) \leq \mathcal{D}(u, x) + \mathcal{D}(x, v)$$

$$\mathcal{D}(u, v) - \mathcal{D}(u, x) \leq \mathcal{D}(x, v)$$

$$\mathcal{D}(u, v) - (\hat{p}_x - p_u) \leq \mathcal{D}(x, v)$$

$$(p_v - p_u) - (\hat{p}_x - p_u) \leq \mathcal{D}(u, v) - (\hat{p}_x - p_u) \leq \mathcal{D}(x, v)$$

$$p_v - \hat{p}_x \leq \mathcal{D}(x, v)$$

Thus the fairness constraint for  $x$  and  $v$  is satisfied, and  $C'$  is an individually fair classifier for  $V \cup \{x\}$ . ◀

Lemma 16 allows us to build up a fair classifier in time  $O(|U|^2)$  from scratch, or to add to an existing fair classifier for a subset. We state several useful corollaries:

► **Corollary 17.** *Given a subset  $V \subset U$  and a classifier  $C : V \times \{0, 1\}^* \rightarrow \{0, 1\}$  such that  $\mathcal{D}(u, v) \geq d(\tilde{C}(u), \tilde{C}(v))$  for all  $u, v \in V$ , there exists an individually fair classifier  $C' : U \times \{0, 1\}^* \rightarrow \{0, 1\}$  which is individually fair for all elements  $u, v \in U$  and has identical behavior to  $C$  on  $V$ .*

Corollary 17 follows immediately from applying Algorithm 4 to each element of  $U \setminus V$  in arbitrary order.

## 33:20 Fairness Under Composition

► **Corollary 18.** *Given a metric  $\mathcal{D}$ , for any pair  $u, v \in U$ , there exists an individually fair classifier  $C : U \times \{0, 1\}^* \rightarrow \{0, 1\}$  such that  $d(\tilde{C}(u), \tilde{C}(v)) = \mathcal{D}(u, v)$ .*

Corollary 18 follows simply from starting from the classifier which is fair only for a particular pair and places them at their maximum distance under  $\mathcal{D}$  and then repeatedly applying Algorithm 4 to the remaining elements of  $U$ . From a distance preservation perspective, this is important; if there is a particular ‘axis’ within the metric where distance preservation is most important, then maximizing the distance between the extremes of that axis can be very helpful for preserving the most relevant distances.

► **Corollary 19.** *Given a metric  $\mathcal{D}$  and  $\alpha \in \mathbb{R}^+$ , for any pair  $u, v \in U$ , there exists an individually fair classifier  $C : U \times \{0, 1\}^* \rightarrow \{0, 1\}$  such that  $p_u/p_v = \alpha$ , where  $p_u = \mathbb{E}[C(u)]$  and likewise  $p_v = \mathbb{E}[C(v)]$ .*

Corollary 19 follows from choosing  $p_u/p_v = \alpha$  without regard for the difference between  $p_u$  and  $p_v$ , and then adjusting. Take  $\beta|p_v - p_u| = \mathcal{D}(u, v)$ , and choose  $\hat{p}_u = \beta p_u$  and  $\hat{p}_v = \beta p_v$  so that  $|\beta p_v - \beta p_u| = \beta|p_v - p_u| \leq \mathcal{D}(u, v)$ , but the ratio  $\frac{\beta p_u}{\beta p_v} = \frac{p_u}{p_v} = \alpha$  remains unchanged.



# A Log-Sobolev Inequality for the Multislice, with Applications

Yuval Filmus<sup>1</sup>

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

Ryan O’Donnell<sup>2</sup>

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

Xinyu Wu

Computer Science Department, Carnegie Mellon University, Pittsburgh, PA, USA  
xinyuw1@andrew.cmu.edu

---

## Abstract

Let  $\kappa \in \mathbb{N}_+^\ell$  satisfy  $\kappa_1 + \dots + \kappa_\ell = n$ , and let  $\mathcal{U}_\kappa$  denote the *multislice* of all strings  $u \in [\ell]^n$  having exactly  $\kappa_i$  coordinates equal to  $i$ , for all  $i \in [\ell]$ . Consider the Markov chain on  $\mathcal{U}_\kappa$  where a step is a random transposition of two coordinates of  $u$ . We show that the log-Sobolev constant  $\varrho_\kappa$  for the chain satisfies

$$\varrho_\kappa^{-1} \leq n \cdot \sum_{i=1}^{\ell} \frac{1}{2} \log_2(4n/\kappa_i),$$

which is sharp up to constants whenever  $\ell$  is constant. From this, we derive some consequences for small-set expansion and isoperimetry in the multislice, including a KKL Theorem, a Kruskal–Katona Theorem for the multislice, a Friedgut Junta Theorem, and a Nisan–Szegedy Theorem.

**2012 ACM Subject Classification** Mathematics of computing → Markov processes

**Keywords and phrases** log-Sobolev inequality, small-set expansion, conductance, hypercontractivity, Fourier analysis, representation theory, Markov chains, combinatorics

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2019.34

**Related Version** Full version at <https://arxiv.org/abs/1809.03546>

## 1 Introduction

Suppose we have a deck of  $n$  cards, with  $\kappa_1$  of them colored red,  $\kappa_2$  of them colored blue, and  $\kappa_3$  of them colored green. If we “shuffle” the cards by repeatedly transposing random pairs of cards, how long does it take for the deck to get to a well-mixed configuration? This question is asking about the mixing time and expansion in a Markov chain known variously as the *multi-urn Bernoulli–Laplace diffusion process* or the *multislice*.

Let  $\ell \in \mathbb{N}_+$  denote a number of *colors* and let  $n \in \mathbb{N}_+$  denote a number of *coordinates* (or *positions*). Following computer science terminology, we refer to elements  $u \in [\ell]^n$  as

---

<sup>1</sup> Taub Fellow – supported by the Taub Foundations. The research was funded by ISF grant 1337/16.

<sup>2</sup> Supported by NSF grant CCF-1717606. This material is based upon work supported by the National Science Foundation under grant numbers listed above. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author and do not necessarily reflect the views of the National Science Foundation (NSF).

*strings.* Given a color  $i \in [\ell]$ , we write  $\#_i u$  for the number of coordinates  $j \in [n]$  for which  $u_j = i$ . The vector  $\kappa = (\#_1 u, \dots, \#_\ell u) \in \mathbb{N}^\ell$  is referred to as the *histogram* of  $u$ . In general, if  $\kappa \in \mathbb{N}_+^\ell$  satisfies  $\kappa_1 + \dots + \kappa_\ell = n$  (so  $\kappa$  is a *composition* of  $n$ ), we define the associated *multislice* to be

$$\mathcal{U}_\kappa = \{u \in [\ell]^n : \#_i u = \kappa_i \text{ for all } i \in [\ell]\}.$$

The terminology here is inspired by the well-studied case when  $\ell = 2$ , in which case  $\mathcal{U}_\kappa$  is a Hamming *slice* of the Boolean cube. We also remark that when  $\ell = n$  and  $\kappa = (1, 1, \dots, 1)$ , the set  $\mathcal{U}_\kappa$  is the set of all permutations of  $[n]$ .

### 1.1 The random transposition Markov chain

The symmetric group  $S_n$  acts on strings  $u \in [\ell]^n$  in the natural way, by permuting coordinates:  $(u^\sigma)_j = u_{\sigma(j)}$  for  $\sigma \in S_n$ . This action preserves each multislice  $\mathcal{U}_\kappa$ . This paper is concerned with the Markov chain on  $\mathcal{U}_\kappa$  generated by *random transpositions*. Let  $\text{Trans}(n) \subseteq S_n$  denote the set of transpositions on  $n$  coordinates. We will specifically be interested in the reversible, discrete-time Markov chain on state space  $\mathcal{U}_\kappa$  in which a step from  $u \in \mathcal{U}_\kappa$  consists of moving to  $u^\tau$ , where  $\tau \sim \text{Trans}(n)$  is chosen uniformly at random. (We always use **boldface** to denote random variables.) One also has the associated *Schreier graph*, with vertex set  $\mathcal{U}_\kappa$  and edges  $\{u, u^\tau\}$  for all  $u \in \mathcal{U}_\kappa$  and  $\tau \in \text{Trans}(n)$ . Since this graph is regular, it follows that the invariant distribution for the Markov chain is the uniform distribution on  $\mathcal{U}_\kappa$ . We will denote this distribution by  $\pi_\kappa$ , or just  $\pi$  if  $\kappa$  is clear from context.

### 1.2 Log-Sobolev inequalities

One of the most powerful ways to study mixing time and “small-set expansion” in Markov chains is through *log-Sobolev* inequalities (see, e.g., [25, 11]). For a subset  $A \subseteq \mathcal{U}_\kappa$ , define its *conductance* (or *expansion*) to be

$$\Phi[A] = \Pr_{\substack{\mathbf{u} \sim A \\ \tau \sim \text{Trans}(n)}} [\mathbf{u}^\tau \notin A].$$

Sets  $A$  with small conductance are natural bottlenecks for mixing in the Markov chain. An example when  $\ell = 2$  and  $\kappa = (n/2, n/2)$  is the “dictator” set  $A = \{u : u_1 = 1\}$ . It has expansion  $\Phi[A] = \frac{1}{n-1}$ , and indeed, if we start the random walk from a string  $u$  with  $u_1 = 1$ , it will take about  $n/2$  steps on average before there’s even a chance that  $u_1$  will change from 1.

One feature of this example is that the set  $A$  is “large”; its (*fractional*) *volume*,

$$\text{vol}(A) = |A|/|\mathcal{U}_\kappa| = \Pr_{\mathbf{u} \sim \pi} [\mathbf{u} \in A],$$

is bounded below by a constant. The “small-set expansion” phenomenon [28, 37, 47] (occurring most famously in the standard random walk on the Boolean cube  $\{0, 1\}^n$ ) refers to the possibility that all “small” sets have high conductance. Intuitively, if small-set expansion holds for a Markov chain, then a random walk with a deterministic starting point should mix rapidly in its early stages, with the possibility for slowdown occurring only when the chain is somewhat close to mixed.

A *log-Sobolev inequality* for the Markov chain is one way that such a phenomenon may be captured. In particular, if the *log-Sobolev constant* for the transposition chain on  $\mathcal{U}_\kappa$  is  $\varrho_\kappa$ , it follows that

$$\Phi[A] \geq \frac{1}{2} \varrho_\kappa \cdot \ln(1/\text{vol}(A)) \quad \text{for all nonempty subsets } A \subseteq \mathcal{U}_\kappa. \tag{1}$$

So sets of constant volume must have conductance  $\Omega(\varrho_\kappa)$ , but sets of volume  $2^{-\Theta(n)}$  (for example) must have conductance  $\Omega(n\varrho_\kappa)$ . A known further consequence of a log-Sobolev inequality is a *hypercontractive inequality*, which concerns expansion in the continuous-time version of the Markov chain. It implies that if  $\sigma$  is the random permutation generated by performing the continuous-time chain for  $t = \frac{\ln c}{2\varrho_\kappa}$  time – i.e.,

$\sigma$  is the product of Poisson  $\left(\frac{\ln c}{2\varrho_\kappa}\right)$  random transpositions,  $c \geq 1$

– then

$$\Pr_{\substack{\mathbf{u} \sim A \\ \sigma \sim \text{Trans}(n)}} [\mathbf{u}^\sigma \notin A] \geq 1 - \text{vol}(A)^{(c-1)/(c+1)} \quad \text{for all nonempty subsets } A \subseteq \mathcal{U}_\kappa.$$

Thus again, if  $\text{vol}(A)$  is small, then the Markov chain will almost surely exit  $A$  after running for  $\Theta(\varrho_\kappa^{-1})$  steps.

We remark that Inequality (1) is merely a *consequence* of the log-Sobolev constant being  $\varrho_\kappa$ . It is not the case that  $\varrho_\kappa$  is defined to be the largest constant for which Inequality (1) holds (for all  $A$ ) – though this is a reasonable intuition. Instead,  $\varrho_\kappa$  is defined to be the largest constant for which a certain generalization of Inequality (1) to nonnegative functions holds; namely,

$$\mathbf{E}_{\substack{\mathbf{u} \sim \pi \\ \tau \sim \text{Trans}(n)}} \left( \sqrt{\phi(\mathbf{u})} - \sqrt{\phi(\mathbf{u}^\tau)} \right)^2 \geq \varrho_\kappa \cdot \text{KL}(\phi\pi \parallel \pi) \quad \text{for all probability densities } \phi. \quad (2)$$

(Here a *probability density function* is a function  $\phi: \mathcal{U}_\kappa \rightarrow \mathbb{R}^{\geq 0}$  satisfying  $\mathbf{E}_\pi[\phi] = 1$ , and  $\text{KL}(\phi\pi \parallel \pi)$  denotes the *KL divergence* between distributions  $\phi\pi$  and  $\pi$ .) Inequality (2) includes Inequality (1) by taking  $\phi = 1_A/\text{vol}(A)$ .

Our main theorem in this work is a lower bound on the log-Sobolev constant for  $\mathcal{U}_\kappa$ :

► **Theorem 1.** *Let  $\kappa \in \mathbb{N}_+^\ell$  satisfy  $\kappa_1 + \dots + \kappa_\ell = n$ , and let  $\varrho_\kappa$  denote the log-Sobolev constant for the transposition chain on the multislice  $\mathcal{U}_\kappa$  (i.e., the largest constant for which Inequality (2) holds). Then*

$$\varrho_\kappa^{-1} \leq n \cdot \sum_{i=1}^{\ell} \frac{1}{2} \log_2(4n/\kappa_i).$$

The main case of interest for us is  $n \rightarrow \infty$  with  $\ell = O(1)$  and  $\kappa_i/n \geq \Omega(1)$  for each  $i$ ; in other words, when we are at a “middling” histogram of a high-dimensional multicube  $[\ell]^n$ . In this case our bound is  $\varrho_\kappa \geq \Omega(1/n)$ , which is the same bound that holds for the standard random walk on the Boolean cube. Thus for this parameter setting, the random transposition chain on the multislice enjoys all of the same small-set expansion properties as the Boolean cube (up to constants).

### 1.3 On the sharpness of Theorem 1

When  $\ell$  is considered to be a constant, Theorem 1 is sharp up to constant factors (which we did not attempt to optimize); i.e.,

$$\varrho_\kappa^{-1} = \Theta(n) \cdot \log \left( \frac{n}{\min_i \{\kappa_i\}} \right) \quad \text{for } \ell = O(1). \quad (3)$$

To see the upper bound on  $\varrho_\kappa$ , assume without loss of generality that  $\ell = \operatorname{argmin}_i \{\kappa_i\}$ , and take

$$A = \{u \in \mathcal{U}_\kappa : u_j = \ell \text{ for all } j \in [\kappa_\ell]\}.$$

It is easy to compute that  $\Phi[A] = \Theta(\kappa_\ell/n)$  and  $\operatorname{vol}(A) = \binom{n}{\kappa_\ell}^{-1}$  (hence  $\ln(1/\operatorname{vol}(A)) = \Theta(\kappa_\ell \log(n/\kappa_\ell))$ ). Putting this into Inequality (1) shows the claimed upper bound on  $\varrho_\kappa$ .

At the opposite extreme, when  $\ell = n$  and  $\kappa = (1, 1, \dots, 1)$ , we have the random transposition walk on the symmetric group  $S_n$ . In this case, Theorem 1 as stated gives the poor bound of  $\varrho_\kappa \geq \Omega(1/n^2 \log n)$ , whereas the optimal bound is  $\varrho_\kappa = \Theta(1/n \log n)$  [11, 35]. In fact, our proof of Theorem 1 (which generalizes that of [35]) can actually achieve the tight lower bound of  $\varrho_\kappa \geq \Omega(1/n \log n)$  in this case. However, we tailored our general bound for the case of  $\ell = O(1)$ , and did not try to optimize for the most general scenario of  $\ell$  varying with  $n$ . A reasonable prediction might be that Equation (3) always holds, up to universal constants, without the assumption of  $\ell = O(1)$ ; we leave investigation of this for future work.

## 2 Applications

There are many known applications of log-Sobolev and hypercontractive inequalities in combinatorics and theoretical computer science (see, e.g., [42, Ch. 9, 10]). In this paper we present four particular consequences of Theorem 1 for analysis/combinatorics of Boolean functions on the multislice. We anticipate the possibility of several more.

### 2.1 KKL and Kruskal–Katona for multislices

Throughout the remainder of this section, let us think of  $n$  as large, of  $\ell$  as constant, and let us fix a histogram  $\kappa$  (with  $\kappa_1 + \dots + \kappa_\ell = n$ ) satisfying  $\kappa_i/n \geq \Omega(1)$  for all  $i$ . For example, we might think of  $\ell = 3$  and  $\kappa = (n/3, n/3, n/3)$ , so that  $\mathcal{U}_\kappa$  consists of all ternary strings with an equal number of 1’s, 2’s, and 3’s. The *isoperimetric problem* for  $\mathcal{U}_\kappa$  would ask: for a given fixed  $0 < \alpha < 1$ , which subset  $A \subseteq \mathcal{U}_\kappa$  with  $\operatorname{vol}(A) = \alpha$  has minimal “edge boundary”, i.e., minimal  $\Phi[A]$ ? (Here “edge boundary” is with respect to performing a single transposition, although in our Kruskal–Katona application we will relate this to the size of  $A$ ’s “shadows” at neighboring multislices.)

We typically think of  $\alpha$  as “constant”, bounded away from 0 and 1. In our example with  $\kappa = (n/3, n/3, n/3)$ , when  $\alpha = 1/3$  the isoperimetric minimizer is a “dictator” set like  $A = \{u : u_1 = 1\}$ ; it has  $\Phi[A] = \frac{4/3}{n-1}$ . The “99% regime” version of the isoperimetric question would be: if  $\Phi[A]$  is within a factor  $1 + o(1)$  of minimal, must  $A$  be “ $o(1)$ -close” to a minimizer? This question will be considered in a companion paper. We will instead consider the “1% regime” version of the isoperimetric question: if  $\Phi[A]$  is at most  $O(1)$  times the minimum, must  $A$  at least “slightly resemble” a minimizer?

To orient ourselves, first note that for constant  $\alpha$  (bounded away from 0 and 1), the minimum possible value of  $\Phi[A]$  among  $A$  with  $\operatorname{vol}(A) = \alpha$  is  $\Theta(1/n)$ ; indeed, this follows from our Theorem 1 and Inequality (1). From this fact, we will derive a multislice variant of the **Kruskal–Katona Theorem**. Up to  $O(1)$  factors, this minimum is achieved not just by “dictator” sets like  $\{u \in \mathcal{U}_{(n/3, n/3, n/3)} : u_1 = 1\}$ , but also by any “junta” set, meaning a set  $A$  for which absence or presence of  $u \in A$  depends only on the colors  $(u_j : j \in J)$  for a set  $J \subseteq [n]$  of cardinality  $c = O(1)$ . It is not hard to see that if  $A \subseteq \mathcal{U}_\kappa$  is such a  $c$ -junta, then  $\Phi[A] \leq O(c/n)$ . We may now ask: if  $\Phi[A] \leq O(1/n)$ , must  $A$  at least slightly “resemble” a junta?

We give two closely related positive answers to this question, as a consequence of our log-Sobolev inequality. The first answer, a **KKL Theorem** for the multislice (cf. [28] and Talagrand’s strengthening of it [51]), follows immediately from previous work [43, 44]. It says that for any set with  $\Phi[A] \leq O(1/n)$ , there must exist some transposition  $\tau \in \text{Trans}(n)$  with at least constant *influence* on  $A$ , where the influence of the transposition  $\tau$  on  $A$  is defined to be

$$\text{Inf}_\tau[A] = \Pr_{\mathbf{u} \sim \pi} [1_A(\mathbf{u}) \neq 1_A(\mathbf{u}^\tau)].$$

Formally, Talagrand’s strengthening of the KKL theorem in this setting is:

► **Theorem 2.** *Let  $f: \mathcal{U}_\kappa \rightarrow \{0, 1\}$ . Then*

$$\text{avg}_{\tau \in \text{Trans}(n)} \left\{ \frac{\text{Inf}_\tau[f]}{\lg(2/\text{Inf}_\tau[f])} \right\} \gtrsim \rho_\kappa \cdot \mathbf{Var}_{\pi_\kappa}[f].$$

Substituting our lower bound on  $\rho_\kappa$  from Theorem 1 yields concrete new results. For example, consider our model scenario of  $n \rightarrow \infty$  with  $\ell = O(1)$  and  $\kappa_i/n \geq \Omega(1)$  for each  $i$ ; suppose further that  $f$  is “roughly balanced”, meaning  $\Omega(1) \leq \mathbf{Var}[f] \leq 1 - \Omega(1)$ . Then

$$\text{avg}_{\tau \in \text{Trans}(n)} \left\{ \frac{\text{Inf}_\tau[f]}{\lg(2/\text{Inf}_\tau[f])} \right\} \gtrsim \frac{1}{n},$$

and hence the maximum influence  $\mathcal{M}[f]$  satisfies

$$\mathcal{M}[f] \gtrsim \frac{\log n}{n}.$$

The latter statement here is the traditional conclusion of the KKL Theorem.

Let us record here one more concrete corollary of Theorem 2. In our model scenario, that theorem (roughly speaking) says that the energy  $\mathcal{E}[1_A] = \text{avg}_{\tau \in \text{Trans}(n)} \text{Inf}_\tau[1_A]$  is at least  $\Omega(\frac{\log n}{n})$  unless some transposition  $(i j)$  has a rather large influence, like  $1/n^{0.1}$ , on  $1_A$ .

► **Corollary 3.** *Let  $A \subseteq \mathcal{U}_\kappa$ . Assume  $\kappa_i \geq pn$  for all  $i \in [\ell]$  and that  $\epsilon \leq \text{vol}(A) \leq 1 - \epsilon$ . Then*

$$\mathcal{E}[1_A] \geq \Omega\left(\frac{\epsilon}{\ell \log(1/p)}\right) \cdot \frac{\log(1/\mathcal{M}[1_A])}{n}.$$

It is the hallmark of a junta  $A$  that every transposition  $\tau$  has either  $\text{Inf}_\tau[A] = 0$  or  $\text{Inf}_\tau[A] \geq \Omega(1)$ . Mirroring the original KKL Theorem, our work shows that: (i) if  $\Phi[A] \leq c/n$  then there exists  $\tau$  with  $\text{Inf}_\tau[A] \geq \exp(-O(c))$ ; (ii) for any  $A \subseteq \mathcal{U}_\kappa$  with  $\Omega(1) \leq \text{vol}(A) \leq 1 - \Omega(1)$ , there exists  $\tau$  with  $\text{Inf}_\tau[A] \geq \Omega(\frac{\log n}{n})$ .

From our KKL Theorem, we obtain various versions of the Kruskal–Katona Theorem for multislices. The classical Kruskal–Katona Theorem [50, 33, 29] concerns subsets of Hamming slices of the Boolean cube. To recall it, let us write a 2-color histogram  $\kappa \in \mathbb{N}_+^2$  as  $(\kappa_0, \kappa_1)$ , with  $n = \kappa_0 + \kappa_1$ . If  $A \subseteq \mathcal{U}_\kappa$ , then the (*lower*) *shadow* of  $A$  is defined to be

$$\partial A = \{v \in \mathcal{U}_{(\kappa_0+1, \kappa_1-1)} : v \leq u \text{ for some } u \in A\}.$$

It is not hard to show that  $\text{vol}(\partial A) \geq \text{vol}(A)$  always (here the fractional volume  $\text{vol}(\partial A)$  is vis-à-vis the containing slice  $\mathcal{U}_{(\kappa_0+1, \kappa_1-1)}$ ). The Kruskal–Katona Theorem improves this by giving an exactly sharp lower bound on  $\text{vol}(\partial A)$  as a function of  $\text{vol}(A)$ . The precise function is somewhat cumbersome to state, but the qualitative consequence, assuming that  $\text{vol}(A)$

and  $\kappa_0/n$  are bounded away from 0 and 1, is that  $\text{vol}(\partial A) \geq \text{vol}(A) + \Omega(1/n)$ . This is sharp, up to the constant in the  $\Omega(\cdot)$ , as witnessed by the “dictator set”  $A = \{u : u_1 = 0\}$ . See [43, Sec. 1.2] for more discussion.

To extend the Kruskal–Katona Theorem to multislices, we first need to extend the notion of neighboring slices and shadows. Fix an ordering on the colors,  $1 \prec 2 \prec \dots \prec \ell$ . This total order extends to a partial order on strings in  $[\ell]^n$  in the natural way.

► **Definition 4.** Let  $\kappa \in \mathbb{N}_+^\ell$  be a histogram. We say that histogram  $\kappa'$  is a *lower neighbor* of  $\kappa$ , and write  $\kappa' \triangleleft \kappa$ , if there exists some  $c \prec d \in [\ell]$  such that  $\kappa'_c = \kappa_c + 1$ ,  $\kappa'_d = \kappa_c - 1$ , and  $\kappa'_i = \kappa_i$  for all other colors  $i$ . In the opposite case, when  $c \succ d$ , we say  $\kappa'$  is an *upper neighbor* of  $\kappa$ , and write  $\kappa' \triangleright \kappa$ .

► **Definition 5.** Let  $A \subseteq \mathcal{U}_\kappa$ , and let  $\kappa' \triangleleft \kappa$ . The *lower shadow* of  $A$  at  $\kappa'$  is

$$\partial_{\kappa'} A = \{u \in \mathcal{U}_{\kappa'} : u \prec v \text{ for some } v \in A\}.$$

We similarly define upper shadows. We may use the same notation  $\partial_{\kappa'} A$  for both kinds of shadows, since whether a shadow is upper or lower is determined by whether  $\kappa' \triangleright \kappa$  or  $\kappa' \triangleleft \kappa$ .

► **Definition 6.** Given a histogram  $\kappa \in \mathbb{N}_+^\ell$ , we define a natural probability distribution  $\mathbf{lower}(\kappa)$  on the lower neighbors of  $\kappa$  as follows. To draw  $\kappa' \sim \mathbf{lower}(\kappa)$ : take an arbitrary  $u \in \mathcal{U}_\kappa$ ; choose  $\mathbf{j}, \mathbf{j}' \sim [n]$  independently and randomly, conditioned on  $u_{\mathbf{j}} \neq u_{\mathbf{j}'}$ ; let  $\mathbf{c}, \mathbf{d}$  denote the two colors  $u_{\mathbf{j}}, u_{\mathbf{j}'}$ , with the convention  $\mathbf{c} \prec \mathbf{d}$ ; finally, let  $\kappa'$  be the lower neighbor of  $\kappa$  with  $\kappa'_c = \kappa_c + 1$  and  $\kappa'_d = \kappa_c - 1$ .

We similarly define a probability distribution  $\mathbf{upper}(\kappa)$  on the upper neighbors of  $\kappa$  by interchanging the roles of  $\mathbf{c}$  and  $\mathbf{d}$ .

► **Theorem 7.** For  $A \subseteq \mathcal{U}_\kappa$  we have

$$\mathbf{E}_{\kappa' \sim \mathbf{lower}(\kappa)} [\text{vol}(\partial_{\kappa'} A)] \geq \text{vol}(A) + \frac{1}{n} \cdot \text{vol}(A) \ln(1/\text{vol}(A)) \cdot \left( \sum_{i=1}^n \log_2(4n/\kappa_i) \right)^{-1}.$$

In particular, at least one lower shadow of  $A$  has volume at least the right-hand side. The analogous statement for upper shadows also holds.

Thus in the model case when  $\text{vol}(A)$  and each  $\kappa_i/n$  is bounded away from 0 and 1, and  $\ell = O(1)$ , we get that the average lower shadow of  $A$  has volume at least  $\text{vol}(A) + \Omega(1/n)$ . Using our KKL Theorem (Corollary 3) we can get a “robust” version of this statement; the volume increase is in fact on the order of  $(\log n)/n$  unless there is a highly influential transposition for  $A$ :

► **Theorem 8.** Let  $A \subseteq \mathcal{U}_\kappa$ . Assume  $\kappa_i \geq pn$  for all  $i \in [\ell]$  and that  $\epsilon \leq \text{vol}(A) \leq 1 - \epsilon$ . Then for any  $\delta > 0$  we have

$$\mathbf{E}_{\kappa' \sim \mathbf{lower}(\kappa)} [\text{vol}(\partial_{\kappa'} A)] \geq \text{vol}(A) + \frac{\log n}{n} \cdot \Omega\left(\frac{\epsilon \delta}{\ell \log(1/p)}\right),$$

or else there exists  $\tau \in \text{Trans}(n)$  with  $\text{Inf}_\tau[A] \geq 1/n^\delta$ . The analogous statement for upper shadows also holds.

As in [43], we give a conceptual improvement to the “or else” clause in Theorem 8. Let us work with upper shadows rather than lower shadows going forward. The natural example for sets  $A$  with upper-shadow expansion “only”  $\Omega(1/n)$  are “dictator” sets such as  $A = \{u : u_1 = \ell\}$ . For such sets, all transpositions of the form  $(1\ j)$  indeed have huge

influence. However, it's not so natural to single out one such  $(1 j)$  as the “reason” for the small expansion; instead, we would prefer to say the reason is that  $A$  is highly “correlated” with coordinate 1:

► **Theorem 9.** *For  $n \rightarrow \infty$ , let  $A \subseteq \mathcal{U}_\kappa$ , with  $\ell = O(1)$ ,  $\kappa_i/n \geq \Omega(1)$  for all  $i \in [\ell]$  and  $\Omega(1) \leq \text{vol}(A) \leq 1 - \Omega(1)$ . Then*

$$\mathbf{E}_{\kappa' \sim \text{upper}(\kappa)}[\text{vol}(\partial_{\kappa'} A)] \geq \text{vol}(A) + \Omega\left(\frac{\log n}{n}\right),$$

or else there exists  $j \in [n]$  and colors  $c \prec d \in [\ell]$  with

$$\Pr_{\mathbf{u} \sim \pi_\kappa}[\mathbf{u} \in A \mid \mathbf{u}_j = d] - \Pr_{\mathbf{u} \sim \pi_\kappa}[\mathbf{u} \in A \mid \mathbf{u}_j = c] \geq 1/n^{0.01}.$$

## 2.2 Friedgut Junta Theorem for multislices

A closely related consequence of our work is a **Friedgut Junta Theorem** for the multislice (cf. [24]), which follows (using a small amount of representation theory) from work of Wimmer [53] (see also [18] for a different account). It states that for any  $A$  with  $\Phi[A] \leq c/n$ , and any  $\epsilon > 0$ , there is a genuine  $\exp(O(c/\epsilon))$ -junta  $A' \subseteq \mathcal{U}_\kappa$  that is  $\epsilon$ -close to  $A$ , meaning  $\text{vol}(A \Delta A') \leq \epsilon$ . The junta theorem can also be generalized to real-valued functions, following the work of Bouyrie [4], with a worse dependence on  $\epsilon$  in the exponent.

► **Theorem 10.** *Let  $f: \mathcal{U}_\kappa \rightarrow \{0, 1\}$  be such that  $\text{Inf}[f] \leq Kn$ . Write  $p_i = \kappa_i/n$ . Then for every  $\epsilon > 0$  there exists  $h: \mathcal{U}_\kappa \rightarrow \{0, 1\}$  depending on at most  $\left(\frac{1}{p_1 p_2 \dots p_\ell}\right)^{O(K/\epsilon)}$  coordinates such that  $\Pr_{\mathbf{u} \sim \pi_\kappa}[f(\mathbf{u}) \neq h(\mathbf{u})] \leq \epsilon$ .*

## 2.3 Nisan–Szegedy Theorem for multislices

Finally, with a little more representation theory effort, we are able to derive from Theorem 1 a **Nisan–Szegedy Theorem** for the multislice (cf. [41]), which is (roughly) an  $\epsilon = 0$  version of the Friedgut Junta Theorem; this generalizes previous work on the Hamming slice [20]. It says that if  $A \subseteq \mathcal{U}_\kappa$  is of “degree  $k$ ” – meaning that its indicator function can be written as a linear combination of  $k$ -junta functions – then  $A$  must be an  $\exp(O(k))$ -junta itself. (The  $k = 1$  case of this theorem, with the conclusion that  $A$  is a 1-junta, was proven recently in [21].)

More formally, the Nisan–Szegedy Theorem says that a degree- $k$  Boolean-valued function on the Hamming cube is a  $k2^k$ -junta. (We remark that the smallest quantity  $\gamma_2(k)$  that can replace  $k2^k$  here is now known [7] to satisfy  $3 \cdot 2^{k-1} - 2 \leq \gamma_2(k) < 22 \cdot 2^k$ .) Let us extend the definition of  $\gamma_2(k)$ ; we'll define  $\gamma_\ell(k)$  to be the least integer such that the following statement is true: Every degree- $k$  Boolean-valued function  $f: [\ell]^n \rightarrow \{0, 1\}$  on the “ $\ell$ -multicube” is a  $\gamma_\ell(k)$ -junta.

Here we say that  $f: [\ell]^n \rightarrow \mathbb{R}$  has degree at most  $k$  if it is a linear combination of  $k$ -juntas (as usual for functions on product spaces, see [42, Def. 8.32]). We can obtain the following Nisan–Szegedy Theorem:

► **Theorem 11.** *There is a universal constant  $C$  such that the following holds. For all  $k \in \mathbb{N}_+$  and all  $\kappa \in \mathbb{N}_+^\ell$  with  $\min_i \{\kappa_i\} \geq \ell^{Ck}$ , if  $f: \mathcal{U}_\kappa \rightarrow \{0, 1\}$  has degree at most  $k$ , then  $f$  is an  $\gamma_\ell(k)$ -junta.*



### 3 Context and prior work

In this section we review similar contexts where log-Sobolev inequalities and small-set expansion have been studied.

#### 3.1 The Boolean cube

The simplest and best-known setting for these kinds of results is the Boolean cube  $\{0, 1\}^n$  with the nearest-neighbour random walk. The optimal hypercontractive inequality in this setting was proven by Bonami [3]. Later, Gross [25] introduced log-Sobolev inequalities, showed that they were equivalent to hypercontractive inequalities in this setting, and determined the exact log-Sobolev constant for the Boolean cube, namely  $\varrho = 2/n$ . Gross also observed that all the same results also hold for Gaussian space in any dimension (recovering prior work of Nelson [40]); Gaussian space is in fact a “special case” of the Boolean cube, by virtue of the Central Limit Theorem. The Boolean cube also generalizes the well-studied *Ehrenfest model* of diffusion [15].

These inequalities for the Boolean cube, as well as the associated small-set expansion corollaries, have had innumerable applications in analysis, combinatorics, and theoretical computer science, in topics ranging from communication complexity to inapproximability; see, e.g., [34] or [42, Chapters 9–11].

A different line of work sought to determine the exact minimum value of  $\Phi[A]$  in terms of the size of  $A$ . This challenge, known as the *edge isoperimetric problem*, has been solved by Harper [26], Lindsey [36], Bernstein [2], and Hart [27], who have shown that the optimal sets are initial segments of a lexicographic ordering of the vertices of the Boolean cube. Recently Ellis, Keller and Lifshitz gave a new proof of the edge isoperimetric inequality using the Kruskal–Katona Theorem [16]. The same set of authors also recently proved a stability version of the edge isoperimetric inequality in the 99% regime [17].

Returning to log-Sobolev inequalities, an extraordinarily helpful feature of the random walk on the Boolean cube is that it is a *product Markov chain*, with a stationary distribution that is *independent* across the  $n$  coordinates. Because of this, a simple induction lets one immediately reduce the log-Sobolev (and hypercontractivity) analysis to the base case of  $n = 1$ .

#### 3.2 Other product chains

For *any* product Markov chain, one can similarly reduce the analysis to the  $n = 1$  case. In general, let  $\nu$  be a probability distribution of full support on  $[\ell]$ , and consider the Markov chain on  $[\ell]^n$  in which a step from  $u \in [\ell]^n$  consists of choosing a random coordinate  $j \sim [n]$  and replacing  $u_j$  with a random draw from  $\nu$ . The invariant distribution for this chain is the product distribution  $\nu^{\otimes n}$ . Though the  $n = 1$  case of this chain is, in a sense, trivial – it mixes perfectly in one step – it is not especially easy to work out the optimal log-Sobolev constant. Nevertheless, Diaconis and Saloffe-Coste [11] showed that for the  $n = 1$  chain, the log-Sobolev constant is

$$\varrho_\nu^{\text{triv}} = 2 \frac{q - p}{\ln q - \ln p}, \quad \text{where } p = \min_{i \in [\ell]} \{\nu(i)\}, \quad q = 1 - p.$$

It follows immediately that the log-Sobolev constant in the general- $n$  case is  $\varrho_\nu^{\text{triv}}/n$ . In particular, if  $\kappa_1 + \dots + \kappa_\ell = n$  and  $\nu(i) = \kappa_i/n$ , then  $\nu^{\otimes n}$  resembles the uniform distribution  $\pi_\kappa$  on  $\mathcal{U}_\kappa$ , and the product chain on  $[\ell]^n$  somewhat resembles the random transposition chain on  $\mathcal{U}_\kappa$ . This gives credence to the possibility that Equation (3) may hold with absolute constants for any  $\ell$ .

### 3.3 The Boolean slice / Bernoulli–Laplace model / Johnson graph

Significant difficulties arise when one moves away from product Markov chains. One of the simplest steps forward is to the Boolean slice. This is the  $\ell = 2$  case of the Markov chains studied in this paper, with the “balanced” case of  $\kappa = (n/2, n/2)$  being the most traditionally studied. This Markov chain is also equivalent to the Bernoulli–Laplace model for diffusion between two incompressible liquids, and to the standard random walk on *Johnson graphs*; taking multiple steps in the chain is similar to the random walk in *generalized Johnson graphs*. The chain has been studied in wide-ranging contexts, from genetics [38], to child psychology [45], to computational learning theory [43]. An asymptotically exact analysis of the time to stationarity of this Markov chain was given by Diaconis and Shahshahani [12], using representation theory. However, the log-Sobolev constant for the chain took a rather long time to be determined; it was left open in Diaconis and Saloff-Coste’s 1996 survey [11] before finally being determined (up to constants) by Lee and Yau in 1998 [35]. This sharp log-Sobolev inequality, and its attendant hypercontractivity and small-set expansion inequalities, have subsequently been used in numerous applications – for the Kruskal–Katona and Erdős–Ko–Rado theorems in combinatorics [43, 8, 22], for computational learning theory [52, 43], for property testing [39], and for generalizing classic “analysis of Boolean functions” results [43, 44, 18, 19, 23, 22, 5].

### 3.4 The Grassmann graph

One direction of generalization for the Johnson graphs are their “ $q$ -analogues”, the *Grassmann graphs*; understanding this Markov chain was posed as an open problem even in the early work of Diaconis and Shahshahani [12, Example 2]. For a finite field  $\mathbb{F}$  and integer parameters  $n \geq k \geq 1$ , the associated Grassmann graph has as its vertices all  $k$ -dimensional subspaces of  $\mathbb{F}^n$ , with two subspaces connected by an edge if their intersection has dimension  $k - 1$ . Understanding small-set expansion (and lack thereof) in the Grassmann graphs was central to the very recent line of work that positively resolved the 2-to-2 Conjecture [31, 14, 13, 1, 32] (with the analogous problems on the Johnson graphs serving as an important warmup [30]). Still, it seems fair to say that the mixing properties of the Grassmann graph are far from being fully understood.

### 3.5 The multislice

We now come to the multislice, the other natural direction of generalization for the Johnson graphs, and the subject of the present paper. One can see the multislice as a generalization of the Bernoulli–Laplace model, modeling diffusion between three or more liquids. As well, the space of functions  $f: \mathcal{U}_\kappa \rightarrow \mathbb{R}$ , together with the action of  $S_n$  on  $\mathcal{U}_\kappa$ , is precisely the *Young permutation module*  $M^\kappa$  arising in the representation theory of the symmetric group. Understanding the mixing properties of the  $\mathcal{U}_\kappa$  Markov chain with random transpositions was suggested as an open problem several times [12], [9, p. 59], [20]. The multislice has also played a key combinatorial role in problems in combinatorics, such as the Density Hales–Jewett problem (where  $\ell = 3$  was the main case under consideration) [46].

Although it might at first appear to be a simple generalization of the Boolean slice, there are several fundamental impediments that arise when moving from  $\ell = 2$  even to  $\ell = 3$ . These include: the fact that a Hamming slice disconnects the nearest-neighbour graph in  $[2]^\ell$  but not in  $[3]^\ell$ ; the fact that one can introduce just *one* variable per coordinate when representing functions  $[2]^\ell \rightarrow \mathbb{R}$  as multilinear polynomials; the fact that 2-row irreps of  $S_n$  (Young diagrams) are completely defined by the number of boxes not in the first

row; and, the fact that when  $\ell \geq 3$ , the decomposition of the permutation module  $M^\kappa$  into irreps has multiplicities. The last of these was the main difficulty to be overcome in Scarabotti's work [48] giving the asymptotic mixing time for the transposition walk on balanced multislices  $\mathcal{U}_{(n/\ell, \dots, n/\ell)}$  (see also [10, 49]). It also prevents the multislice from forming an association scheme.

For the purposes of this paper, the main difficulty that arises when analyzing the log-Sobolev inequality is the following: when  $\ell = 2$ , any nontrivial step in the Markov chain (switching a 1 and a 2) has the property that the histogram within  $[\ell]^{n-2}$  of the unswitched colors is always the same:  $(\kappa_1 - 1, \kappa_2 - 1)$ . By contrast, once  $\ell \geq 3$ , the multiple “kinds” of transpositions (switching a 1 and a 2, or a 1 and a 3, or a 2 and a 3, etc.) lead to differing histograms within  $[\ell]^{n-2}$  for the unswitched colors. This significantly complicates inductive arguments.

### 3.6 The symmetric group and beyond

Finally, we mention that analysis of the multislice can also be motivated simply as a necessary first step in a full understanding of spectral analysis on the symmetric group and other algebraic structures, an opinion also espoused in, e.g., [6]. Such structures include classical association schemes such as polar spaces and bilinear forms, matrix groups such as the general linear group, and the  $q$ -analog of the multislice.

---

#### References

- 1 Boaz Barak, Pravesh Kothari, and David Steurer. Small-Set Expansion in Shortcode Graph and the 2-to-2 Conjecture. Technical Report 1804.08662, arXiv, 2018.
- 2 Arthur Jay Bernstein. Maximally connected arrays on the  $n$ -cube. *SIAM J. Appl. Math.*, 15:1485–1489, 1967. doi:10.1137/0115129.
- 3 Aline Bonami. Étude des coefficients Fourier des fonctions de  $L^p(G)$ . *Annales de l'Institut Fourier*, 20(2):335–402, 1970.
- 4 Raphaël Bouyrie. An unified approach to the Junta theorem for discrete and continuous models. Technical report, arXiv, 2017. arXiv:1702.00753.
- 5 Raphaël Bouyrie. On quantitative noise stability and influences for discrete and continuous models. *Combin. Probab. Comput.*, 27(3):334–357, 2018. doi:10.1017/S0963548318000044.
- 6 Sourav Chatterjee, Jason Fulman, and Adrian Röllin. Exponential approximation by Stein's method and spectral graph theory. *ALEA Lat. Am. J. Probab. Math. Stat.*, 8:197–223, 2011.
- 7 John Chiarelli, Pooya Hatami, and Michael Saks. Tight Bound on the Number of Relevant Variables in a Bounded degree Boolean function. Technical report, arXiv, 2018. arXiv:1801.08564.
- 8 Pat Devlin and Jeff Kahn. On “stability” in the Erdős-Ko-Rado theorem. *SIAM J. Discrete Math.*, 30(2):1283–1289, 2016. doi:10.1137/15M1012992.
- 9 Persi Diaconis. *Group representations in probability and statistics*, volume 11 of *Institute of Mathematical Statistics Lecture Notes—Monograph Series*. Institute of Mathematical Statistics, Hayward, CA, 1988.
- 10 Persi Diaconis and Susan Holmes. Random walks on trees and matchings. *Electron. J. Probab.*, 7:no. 6, 17, 2002. doi:10.1214/EJP.v7-105.
- 11 Persi Diaconis and Laurent Saloff-Coste. Logarithmic Sobolev inequalities for finite Markov chains. *Annals of Applied Probability*, 6(3):695–750, 1996.
- 12 Persi Diaconis and Mehrdad Shahshahani. Time to reach stationarity in the Bernoulli–Laplace diffusion model. *SIAM Journal on Mathematical Analysis*, 18(1):208–218, 1987.

- 13 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 Symposium on Theory of Computing*, pages 940–951, 2018.
- 14 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 Symposium on Theory of Computing*, pages 376–389, 2018.
- 15 Paul Ehrenfest and Tatiana Ehrenfest. Über zwei bekannte Einwände gegen das Boltzmannsche  $H$ -Theorem. *Physikalische Zeitschrift*, 8(9):311–314, 1907.
- 16 David Ellis, Nathan Keller, and Noam Lifshitz. On a biased edge isoperimetric inequality for the discrete cube. Technical report, arXiv, 2017. [arXiv:1702.01675](https://arxiv.org/abs/1702.01675).
- 17 David Ellis, Nathan Keller, and Noam Lifshitz. On the structure of subsets of the discrete cube with small edge boundary. *Discrete Analysis*, 9:1–29, 2018. [doi:10.19086/da.3668](https://doi.org/10.19086/da.3668).
- 18 Yuval Filmus. An orthogonal basis for functions over a slice of the Boolean hypercube. *Electron. J. Combin.*, 23(1):Paper 1.23, 27, 2016.
- 19 Yuval Filmus. Friedgut–Kalai–Naor theorem for slices of the Boolean cube. *Chic. J. Theoret. Comput. Sci.*, pages Art. 14, 17, 2016.
- 20 Yuval Filmus and Ferdinand Ihringer. Boolean constant degree functions on the slice are juntas. Technical report, arXiv, 2018. [arXiv:1801.06338](https://arxiv.org/abs/1801.06338).
- 21 Yuval Filmus and Ferdinand Ihringer. Boolean degree 1 functions on some classical association schemes. Technical report, arXiv, 2018. [arXiv:1801.06034](https://arxiv.org/abs/1801.06034).
- 22 Yuval Filmus, Guy Kindler, Elchanan Mossel, and Karl Wimmer. Invariance principle on the slice. *Transactions on Computation Theory*, 10(3):11, 2018.
- 23 Yuval Filmus and Elchanan Mossel. Harmonicity and invariance on slices of the Boolean cube. In *Proceedings of the 31st Annual Computational Complexity Conference*, pages Art. No. 16, 13, 2016.
- 24 Ehud Friedgut. Boolean functions with low average sensitivity depend on few coordinates. *Combinatorica*, 18(1):27–36, 1998.
- 25 Leonard Gross. Logarithmic Sobolev inequalities. *American Journal of Mathematics*, 97(4):1061–1083, 1975.
- 26 Lawrence H. Harper. Optimal assignments of numbers to vertices. *J. Soc. Indust. Appl. Math.*, 12:131–135, 1964.
- 27 Sergiu Hart. A note on the edges of the  $n$ -cube. *Discrete Math.*, 14(2):157–163, 1976. [doi:10.1016/0012-365X\(76\)90058-3](https://doi.org/10.1016/0012-365X(76)90058-3).
- 28 Jeff Kahn, Gil Kalai, and Nathan Linial. The influence of variables on Boolean functions. In *Proceedings of the 29th Annual IEEE Symposium on Foundations of Computer Science*, pages 68–80, 1988.
- 29 Gyula Katona. A theorem of finite sets. In *Theory of graphs (Proc. Colloq., Tihany, 1966)*, pages 187–207. Academic Press, New York, 1968.
- 30 Subhash Khot, Dor Minzer, Dana Moshkovitz, and Muli Safra. Small Set Expansion in The Johnson Graph. Technical Report TR18-078, ECCC, 2018.
- 31 Subhash Khot, Dor Minzer, and Muli Safra. On independent sets, 2-to-2 games, and Grassmann graphs. In *Proceedings of the 49th Annual ACM Symposium on Theory of Computing*, pages 576–589, 2017.
- 32 Subhash Khot, Dor Minzer, and Muli Safra. Pseudorandom sets in Grassmann graph have near-perfect expansion. Technical Report TR18-006, ECCC, 2018.
- 33 Joseph B. Kruskal. The number of simplices in a complex. In *Mathematical optimization techniques*, pages 251–278. Univ. of California Press, Berkeley, Calif., 1963.
- 34 Michel Ledoux. Concentration of measure and logarithmic Sobolev inequalities. In *Séminaire de Probabilités XXXIII*, pages 120–216. Springer, 1999.

- 35 Tzong-Yau Lee and Horng-Tzer Yau. Logarithmic Sobolev inequality for some models of random walks. *Annals of Probability*, 26(4):1855–1873, 1998.
- 36 John H. Lindsey II. Assignment of numbers to vertices. *Amer. Math. Monthly*, 71:508–516, 1964. doi:10.2307/2312587.
- 37 László Lovász and Ravi Kannan. Faster mixing via average conductance. In *Proceedings of the 31st Annual ACM Symposium on Theory of Computing*, pages 282–287, 1999.
- 38 Patrick Moran. Random processes in genetics. *Mathematical Proceedings of the Cambridge Philosophical Society*, 54(1):60–71, 1958.
- 39 Dana Moshkovitz. Direct Product Testing With Nearly Identical Sets. Technical Report TR14-182, ECCO, 2014.
- 40 Edward Nelson. The free Markoff field. *Journal of Functional Analysis*, 12:211–227, 1973.
- 41 Noam Nisan and Mario Szegedy. On the Degree of Boolean Functions as Real Polynomials. *Computational Complexity*, 4(4):301–313, 1994.
- 42 Ryan O’Donnell. *Analysis of Boolean Functions*. Cambridge University Press, 2014.
- 43 Ryan O’Donnell and Karl Wimmer. KKL, Kruskal-Katona, and monotone nets. *SIAM Journal on Computing*, 42(6):2375–2399, 2013.
- 44 Ryan O’Donnell and Karl Wimmer. Sharpness of KKL on Schreier graphs. *Electronic Communications in Probability*, 18:1–12, 2013.
- 45 Jean Piaget and Barbel Inhelder. *The origin of the idea of chance in children*. The Norton Library, 1976.
- 46 D. H. J. Polymath. A new proof of the density Hales–Jewett theorem. *Annals of Mathematics*, 175(3):1283–1327, 2012.
- 47 Prasad Raghavendra and David Steurer. Graph expansion and the Unique Games Conjecture. In *Proceedings of the 42nd Annual ACM Symposium on Theory of Computing*, pages 755–764, 2010.
- 48 Fabio Scarabotti. Time to reach stationarity in the Bernoulli–Laplace diffusion model with many urns. *Adv. in Appl. Math.*, 18(3):351–371, 1997. doi:10.1006/aama.1996.0514.
- 49 Fabio Scarabotti and Filippo Tolli. Harmonic analysis on a finite homogeneous space II: the Gelfand–Tsetlin decomposition. *Forum Math.*, 22(5):879–911, 2010. doi:10.1515/FORUM.2010.047.
- 50 Marcel-Paul Schützenberger. A characteristic property of certain polynomials of E. F. Moore and C. E. Shannon. *Quarterly Progress Report, Research Laboratory of Electronics (RLE)*, 055.IX:117–118, 1959.
- 51 Michel Talagrand. On Russo’s approximate zero-one law. *Annals of Probability*, 22(3):1576–1587, 1994.
- 52 Karl Wimmer. *Fourier methods and combinatorics in learning theory*. PhD thesis, Carnegie Mellon University, 2009.
- 53 Karl Wimmer. Low influence functions over slices of the Boolean hypercube depend on few coordinates. In *Proceedings of the 29th Annual Computational Complexity Conference*, pages 120–131, 2014.

# Cubic Formula Size Lower Bounds Based on Compositions with Majority

Anna Gál<sup>1</sup>


The University of Texas at Austin, Austin, TX, USA  
panni@cs.utexas.edu

Avishay Tal<sup>2</sup>

Stanford University, Palo Alto, CA, USA  
avishay.tal@gmail.com

Adrian Trejo Nuñez

The University of Texas at Austin, Austin, TX, USA  
atrejo@cs.utexas.edu

 <https://orcid.org/0000-0002-5658-9956>

---

## Abstract

We define new functions based on the Andreev function and prove that they require  $n^3/\text{polylog}(n)$  formula size to compute. The functions we consider are generalizations of the Andreev function using compositions with the majority function. Our arguments apply to composing a hard function with any function that agrees with the majority function (or its negation) on the middle slices of the Boolean cube, as well as iterated compositions of such functions. As a consequence, we obtain  $n^3/\text{polylog}(n)$  lower bounds on the (non-monotone) formula size of an explicit monotone function by combining the monotone address function with the majority function.

**2012 ACM Subject Classification** Theory of computation → Computational complexity and cryptography, Theory of computation → Circuit complexity

**Keywords and phrases** formula lower bounds, random restrictions, KRW conjecture, composition

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2019.35

**Acknowledgements** We thank an anonymous referee of a previous version of this paper for suggesting to apply our results to prove cubic formula size lower bounds for a monotone function.

## 1 Introduction

We study the problem of proving lower bounds on the De Morgan formula size of explicit functions.

While it is known that almost all Boolean functions of  $n$  variables require formula size exponential in  $n$ , proving lower bounds on the formula size of specific functions remains a major challenge. The current largest lower bounds on De Morgan formula size for explicitly defined functions are of the form  $n^{3-o(1)}$ . Lower bounds for general formula size are weaker, throughout this paper we only consider De Morgan formulas, but sometimes we just refer to them as “formulas”.

---

<sup>1</sup> Part of this work was done while visiting the Simons Institute for the Theory of Computing.

<sup>2</sup> Supported by a Motwani Postdoctoral Fellowship and by NSF grant CCF-1763299. Part of this work was done while visiting the Simons Institute for the Theory of Computing.



## History

Formula size lower bounds have a long history. One of the methods for proving formula size lower bounds is based on shrinkage of De Morgan formulas under random restrictions. This method was introduced by Subbotovskaya [17] who gave a  $\Omega(n^{1.5})$  lower bound on the De Morgan formula size of the parity function. The lower bound for parity has been improved by Khrapchenko [10] to  $\Omega(n^2)$ . However, it is also known that Khrapchenko's method cannot give larger than quadratic lower bounds. The method of random restrictions on the other hand has led to the currently known largest lower bounds on formula size. Andreev [1] used random restrictions to prove an  $\Omega(n^{2.5-o(1)})$  lower bound for a function obtained by composing parity with an arbitrary other function  $f$  where  $f$  is specified as part of the input. We give more formal definitions in Section 2. After improvements of the bound by [8, 13], Håstad [5] proved a lower bound of the form  $n^{3-o(1)}$  for the Andreev function. Tal [18] improved the lower order terms to give a  $\Omega\left(\frac{n^3}{(\log n)^2 \log \log n}\right)$  lower bound for the Andreev function, which is tight up to the  $\log \log n$  term. Tal [19] gave a slightly larger lower bound of the form  $\Omega\left(\frac{n^3}{\log n (\log \log n)^2}\right)$  for another function introduced by Komargodski and Raz [11]. This function is similar to the Andreev function, it still composes parity with other functions specified as part of the input. The difference is that instead of specifying the function  $f$  by its entire truth table as part of the input, an error correcting code is used to derive the truth table from the input. Bogdanov [2] showed that the same  $\Omega\left(\frac{n^3}{\log n (\log \log n)^2}\right)$  lower bound can also be obtained for any “small-biased” function, that is any randomized function whose distribution of truth tables is small biased. He also noted that standard constructions of small biased sets yield explicit families of such functions. [3, 12] showed that parity in Andreev's function can be replaced with any good enough bit fixing extractor, and the resulting function still requires  $n^3/\text{polylog}(n)$  formula size.

Other than Bogdanov's functions, the only explicit function with  $n^{3-o(1)}$  formula size lower bounds has been the Andreev function, and its variants using error correcting codes by [11, 19] or bit fixing extractors [3, 12].

Dinur and Meir [4] gave a new proof of  $n^{3-o(1)}$  formula size lower bounds for the Andreev function, based on information theoretic arguments. The bound obtained by their argument is of the form  $\Omega\left(\frac{n^3}{2^{\sqrt{\log n} \text{poly} \log \log n}}\right)$  which is weaker in the lower order terms than the bounds of Håstad [5] and Tal [18]. But their goal was to give a proof that could possibly generalize to other function compositions, which would be important in light of the KRW conjecture [9] (see Section 5). Our results can be viewed as a step in this direction.

## Our Results

In this paper we obtain  $n^3/\text{polylog}(n)$  lower bounds on a new class of functions. First we consider an extension of the Andreev function which we call “Generalized Andreev function with Majority”, using the majority function instead of parity in the function compositions. We define the function formally in Section 2. As far as we know this function has not been studied before, and previous approaches do not directly work to obtain our bounds.

Next we extend our results to composing a hard function with any function that agrees with the majority function (or its negation) on the middle slices of the Boolean cube, as well as iterated compositions of such functions. Since parity agrees with majority on the two middle slices of the Boolean cube, our argument also applies to parity (the original Andreev function), and composing parity with majority in various ways.



As another consequence, we prove  $n^3/\text{polylog}(n)$  lower bounds on the (non-monotone) formula size of the monotone function obtained by combining the monotone address function of Wegener [21] with the majority function.

It was pointed out to us by Pavel Pudlak [14], that for any function  $f : \{0, 1\}^n \rightarrow \{0, 1\}$ , one can construct a function  $f' : \{0, 1\}^{2n} \rightarrow \{0, 1\}$  that is monotone, and has formula size at least as large as  $f$ . Consider inputs of the form  $(x, y)$  where  $x, y \in \{0, 1\}^n$  and simply let  $f'(x, y) = 1$  on all inputs of Hamming weight greater than  $n$ ,  $f'(x, y) = 0$  on all inputs of Hamming weight less than  $n$ , and  $f'(x, y) = f(x)$  on inputs  $(x, y)$  with Hamming weight  $n$ . Observe that  $f'$  has formula size at least as large as  $f$ : identifying for each  $i \in \{1, \dots, n\}$  the literals  $x_i$  and  $\neg y_i$ , and similarly  $\neg x_i$  and  $y_i$ , we get  $f$  from  $f'$ . However, as far as we know, our results give the first super-quadratic formula size lower bound with a direct proof for an explicitly defined monotone function.

Our argument gives a formal proof that the monotone formula size of the majority function is at least  $n^{\Gamma_{\text{mon}}}/\text{polylog } n$ , where  $\Gamma_{\text{mon}}$  denotes the shrinkage exponent of monotone formulas under random restrictions. It is a long standing open problem to determine the value of  $\Gamma_{\text{mon}}$ . It is also open to obtain tight bounds on the formula size of majority, both in the monotone and non-monotone case. The current best lower bound for both monotone and non-monotone formulas computing majority of  $n$  bits is  $\Omega(n^2)$ . The best upper bound on the De Morgan formula size of majority on  $n$  bits is  $\mathcal{O}(n^{3.91})$  [16]. Considering monotone formulas for majority, the best upper bound remains the  $\mathcal{O}(n^{5.3})$  bound by Valiant [20]. Håstad [5] noted that determining the value of  $\Gamma_{\text{mon}}$  is likely to yield improved lower bounds on the monotone formula size of the majority function. Our results make this connection explicit, independently of how the value  $\Gamma_{\text{mon}}$  is obtained.

Our argument is based on random restrictions and analyzing the shrinkage of formula size under restrictions. However, the main obstacle in applying previous arguments is that we need random restrictions that leave each Majority undetermined. Previously considered restrictions are far from achieving this. Instead of standard random restrictions, we use “staged” random restrictions, and adjust their results to enforce more structure. The idea of building restrictions in stages appears before in [7, 3, 12]. The main difference in our approach is that we maintain the structure of the composed hard function with majority after each stage by performing some clean-up procedure.

In addition to worst case formula size lower bounds, average case lower bounds have been shown in [3, 11, 12, 19]. These bounds are quantitatively weaker than the  $n^{3-o(1)}$  worst case bounds but provide high probability versions of the shrinkage results under certain structured random reductions. Tal [19] has shown that average case bounds can be used to obtain stronger worst case bounds, and in fact the current largest lower bounds of the form  $\Omega\left(\frac{n^3}{\log n(\log \log n)^2}\right)$  by Tal [19] and Bogdanov [2] were obtained this way.

## 2 Definitions and Background

Given an  $n$ -bit string  $\vec{x} = (x_1, \dots, x_n) \in \{0, 1\}^n$ , let  $\text{wt}(\vec{x})$  denote the Hamming weight of  $\vec{x}$ , defined as

$$\text{wt}(\vec{x}) = |\{i : x_i = 1\}|$$

Let  $\mathcal{B}_n = \{f : \{0, 1\}^n \rightarrow \{0, 1\}\}$  denote the set of all Boolean functions on  $n$  bits.

Given a  $bm$ -bit string  $\vec{x}$ , we can interpret  $\vec{x}$  as a  $b \times m$  matrix with rows  $\vec{x}_1, \dots, \vec{x}_b$  of  $m$  bits each. If  $f \in \mathcal{B}_b$  and  $g \in \mathcal{B}_m$  are arbitrary functions, let  $f \circ g : \{0, 1\}^{b \times m} \rightarrow \{0, 1\}$  denote

their composition, defined as

$$(f \circ g)(\vec{x}_1, \dots, \vec{x}_b) = f(g(\vec{x}_1), \dots, g(\vec{x}_b))$$

Given a function  $f \in \mathcal{B}_n$ , let  $\text{tt}(f)$  denote the truth table of  $f$ , defined as the string of length  $2^n$  specifying the output of  $f$  on all strings  $\vec{x} \in \{0, 1\}^n$  in lexicographic order. We use  $f$  and  $\text{tt}(f)$  interchangeably when  $f$  is an input to another function.

Let  $\oplus_m : \{0, 1\}^m \rightarrow \{0, 1\}$  denote the parity function on  $m$  bits.

Let  $\text{Maj}_m : \{0, 1\}^m \rightarrow \{0, 1\}$  denote the majority function on  $m$  bits, defined as

$$\text{Maj}_m(\vec{x}) = \begin{cases} 1 & \text{if } \text{wt}(\vec{x}) \geq \lceil \frac{m}{2} \rceil \\ 0 & \text{otherwise} \end{cases}$$

## 2.1 Andreev Function

Let  $\mathcal{A}_n : \{0, 1\}^n \times \{0, 1\}^n \rightarrow \{0, 1\}$  denote the Andreev function on  $2n$  bits. Let  $b = \log n$  and  $m = n/b = n/\log n$ . If  $f \in \mathcal{B}_b$ , then  $|\text{tt}(f)| = 2^b = 2^{\log n} = n$ .

The function  $\mathcal{A}_n$  takes two inputs: an  $n$ -bit string representing the truth table of a function  $f$  on  $b$  bits, and an  $n$ -bit string  $\vec{x}$ , interpreted as a  $b \times m$  matrix with rows  $\vec{x}_1, \dots, \vec{x}_b$ . Then,

$$\mathcal{A}_n(f, \vec{x}) = (f \circ \oplus_m)(\vec{x}) = f(\oplus_m(\vec{x}_1), \dots, \oplus_m(\vec{x}_b))$$

## 2.2 Generalized Andreev Function

Let  $b = \log n$  and  $m = n/b$  as before. If  $g_m \in \mathcal{B}_m$  is an arbitrary function on  $m$  bits, then let  $\mathcal{A}_n^{g_m} : \{0, 1\}^n \times \{0, 1\}^n \rightarrow \{0, 1\}$  denote the generalized Andreev function on  $2n$  bits, defined analogously by

$$\mathcal{A}_n^{g_m}(f, \vec{x}) = (f \circ g_m)(\vec{x}) = f(g_m(\vec{x}_1), \dots, g_m(\vec{x}_b))$$

In particular,  $\mathcal{A}_n = \mathcal{A}_n^{\oplus_m}$ .

Let  $\mathcal{M}_n : \{0, 1\}^n \times \{0, 1\}^n \rightarrow \{0, 1\}$  denote the generalized Andreev function with  $\text{Maj}_m$  in place of  $g_m$ . That is

$$\mathcal{M}_n(f, \vec{x}) = \mathcal{A}_n^{\text{Maj}_m}(f, \vec{x}) = (f \circ \text{Maj}_m)(\vec{x}) = f(\text{Maj}_m(\vec{x}_1), \dots, \text{Maj}_m(\vec{x}_b))$$

If  $f \in \mathcal{B}_b$  is a fixed function, define  $\mathcal{M}_{n,f} : \{0, 1\}^n \rightarrow \{0, 1\}$  as

$$\mathcal{M}_{n,f}(\vec{x}) = \mathcal{M}_n(f, \vec{x})$$

or equivalently,  $\mathcal{M}_{n,f} = f \circ \text{Maj}_m$ .

## 2.3 De Morgan Formulas

Formulas are tree-like circuits, that is circuits where each gate has fan-out at most one. A De Morgan formula is a formula that uses only **AND**, **OR** and negation gates, where the gates have fan-in at most 2. Let  $f \in \mathcal{B}_n$  be an arbitrary function. Define  $\mathcal{L}(f)$  to be the formula complexity of  $f$ , which is the minimum number of leaves required by any De Morgan formula computing  $f$ .

It is known that almost all Boolean functions on  $n$  variables require De Morgan formula size at least  $\frac{2^n}{2 \log n}$  [15].

## 2.4 Random Restrictions and Shrinkage

Consider a function  $f \in \mathcal{B}_n$  and let  $S = \{x_1, \dots, x_n\}$  denote the variables of  $f$ . A restriction on  $S$  is a function  $\rho: S \rightarrow \{0, 1, \star\}$ . Let  $f \upharpoonright_\rho$  denote the function obtained from  $f$  by fixing inputs  $x_i$  to  $\rho(x_i)$  if  $\rho(x_i) \neq \star$ , which depends only on the inputs  $x_i$  for which  $\rho(x_i) = \star$ . Given arbitrary functions  $g \in \mathcal{B}_m$  and  $f \in \mathcal{B}_n$  for  $m \leq n$ , we say that  $f$  computes  $g$  as a sub-function if  $g$  can be achieved as a restriction of  $f$ .

A random  $p$ -restriction on  $S$  is a randomly generated restriction  $\rho$  where

$$\begin{aligned} \Pr(\rho(x_i) = \star) &= p \\ \Pr(\rho(x_i) = 0) &= \Pr(\rho(x_i) = 1) = \frac{1-p}{2} \end{aligned}$$

uniformly and independently for all  $x_i \in S$ . Let  $\mathcal{R}_p$  denote the distribution of all uniformly generated random  $p$ -restrictions.

Subbotovskaya [17] proved that for any Boolean function  $f \in \mathcal{B}_n$  it holds that

$$\mathbb{E}_{\rho \sim \mathcal{R}_p} \left[ \mathcal{L}(f \upharpoonright_\rho) \right] = \mathcal{O}(p^\Gamma \mathcal{L}(f))$$

for  $\Gamma = 3/2$ . The constant  $\Gamma$  is called the shrinkage exponent, which is the largest number for which the statement is true. After several improvements [8, 13], Håstad [5] proved that  $\Gamma = 2$ . The following version is due to Tal [18].

► **Theorem 2.1** (Shrinkage Lemma [18]). *Let  $f \in \mathcal{B}_n$  be an arbitrary function. Then,  $\forall p > 0$ ,*

$$\mathbb{E}_{\rho \sim \mathcal{R}_p} \left[ \mathcal{L}(f \upharpoonright_\rho) \right] \leq \mathcal{O}(1 + p^2 \mathcal{L}(f)) \quad (1)$$

► **Corollary 2.2.** *Let  $f \in \mathcal{B}_n$  be an arbitrary function. Then,  $\exists c > 0$  such that for  $\forall p > 0$  and large enough  $n$ ,*

$$\Pr_{\rho \sim \mathcal{R}_p} \left( \mathcal{L}(f \upharpoonright_\rho) \geq 10c(1 + p^2 \mathcal{L}(f)) \right) \leq \frac{1}{10} \quad (2)$$

**Proof.** Let  $c > 0$  be chosen such that  $\mathbb{E}_{\rho \sim \mathcal{R}_p} \left[ \mathcal{L}(f \upharpoonright_\rho) \right] \leq c(1 + p^2 \mathcal{L}(f))$ . Then, by Markov's inequality:  $\Pr_{\rho \sim \mathcal{R}_p} \left( \mathcal{L}(f \upharpoonright_\rho) \geq 10c(1 + p^2 \mathcal{L}(f)) \right) \leq \frac{1}{10}$ . ◀

## 2.5 Concentration Inequalities

We use the following result on bounds of sums of random variables.

► **Theorem 2.3** (Hoeffding's Inequality [6]). *Let  $X_1, \dots, X_n$  be independent random variables such that  $a_i \leq X_i \leq b_i$  for  $1 \leq i \leq n$  and let  $X = \sum_{i=1}^n X_i$ . Then,*

$$\Pr \left( \left| X - \mathbb{E}[X] \right| \geq t \right) \leq 2 \exp \left( \frac{-2t^2}{\sum_{i=1}^n (b_i - a_i)^2} \right) \quad (3)$$

## 3 Composition with Majority

Let  $\mathcal{M}_n$  be the generalized Andreev function with majority. Let  $b = \log n$  and  $m = n/b = n/\log n$  and assume  $b, m \in \mathbb{N}$ . Let  $h \in \mathcal{B}_b$  be a function of maximum formula complexity and consider  $\mathcal{M}_{n,h} = h \circ \text{Maj}_m$ . Since  $\mathcal{M}_{n,h}$  is a sub-function of  $\mathcal{M}_n$ , we

have  $\mathcal{L}(\mathcal{M}_n) \geq \mathcal{L}(\mathcal{M}_{n,h}) = \mathcal{L}(h \circ \text{Maj}_m)$ . Thus, it suffices to prove a lower bound on the formula complexity of  $h \circ \text{Maj}_m$ . Indeed, this will be our strategy (which is standard when proving lower bounds for Andreev-type functions). Our main result is the following general theorem, that may also be applied in other scenarios.

► **Theorem 3.1** (Formula Size of Composition with Majority). *Let  $b, m \in \mathbb{N}$  and  $h \in \mathcal{B}_b$  be non-constant. Then,*

$$\mathcal{L}(h \circ \text{Maj}_m) \geq \mathcal{L}(h) \cdot m^2 / \text{polylog}(b \cdot m)$$

Since the hardest functions on  $b = \log n$  bits have formula complexity at least  $\frac{n}{2 \log \log n}$  [15], Theorem 3.1 implies that

$$\mathcal{L}(\mathcal{M}_n) \geq \mathcal{L}(h \circ \text{Maj}_m) \geq \mathcal{L}(h) \cdot m^2 / \text{polylog}(b \cdot m) \geq n^3 / \text{polylog}(n)$$

The rest of this section is devoted to the proof of Theorem 3.1.

### Warmup

Let  $n = mb$ . The input  $\vec{x} = (x_1, \dots, x_n)$  is divided into  $b$  contiguous blocks  $B_1, \dots, B_b$  of  $m$  variables each. In order to apply a random restriction based argument to  $h \circ \text{Maj}_m$ , we wish to prove that there exists a restriction  $\rho$  that leaves each  $\text{Maj}_m$  undetermined and the resulting formula shrinks by a factor of  $\Omega(m^2 / \text{polylog}(bm))$ .

A single majority is left undetermined by  $\rho$  if the absolute difference between the number of variables assigned 0 and 1 is at most the number of unassigned variables. Otherwise, the majority value is already set and there are not enough remaining variables to flip it.

### 3.1 Random $p$ -Restrictions

Previous random restriction based arguments typically use random  $p$ -restrictions defined in Section 2.4. We start by some observations about them. Let  $\rho \in \mathcal{R}_p$  be a random  $p$ -restriction on  $S = \{x_1, \dots, x_n\}$  and let  $B_i = \{x_{i_1}, \dots, x_{i_m}\}$  be a fixed block of the input.

Let  $X_{ik}$  and  $Y_{ik}$  for  $1 \leq k \leq m$  be the following random variables:

$$X_{ik} = \begin{cases} 1 & \text{if } \rho(x_{i_k}) = \star \\ 0 & \text{otherwise} \end{cases} \quad Y_{ik} = \begin{cases} 1 & \text{if } \rho(x_{i_k}) = 0 \\ -1 & \text{if } \rho(x_{i_k}) = 1 \\ 0 & \text{if } \rho(x_{i_k}) = \star \end{cases}$$

Then,

$$\begin{aligned} X_i &= \sum_{k=1}^m X_{ik} & Y_i &= \sum_{k=1}^m Y_{ik} \\ \mathbb{E}[X_i] &= \sum_{k=1}^m \mathbb{E}[X_{ik}] = mp & \mathbb{E}[Y_i] &= \sum_{k=1}^m \mathbb{E}[Y_{ik}] = 0 \end{aligned}$$

We note that  $X_i$  and  $Y_i$  are not necessarily independent, since for any  $\ell \geq 0$

$$X_i \geq \ell \implies |Y_i| \leq m - \ell$$

Since  $0 \leq X_{ik} \leq 1$ , Theorem 2.3 gives:

$$\Pr_{\rho \in \mathcal{R}_p} (|X_i - mp| > t) \leq 2 \exp(-2t^2/m)$$

To obtain lower bounds of the form  $\mathcal{L}(h) \cdot \frac{m^2}{\text{polylog}(mb)}$  by a single round of  $p$ -restrictions, one would need  $p = \mathcal{O}(\text{polylog}(mb)/m)$ , thus  $X_i = \Theta(\text{polylog}(mb))$  would hold with high probability. Since  $|Y_i|$  is typically  $\Omega(\sqrt{m})$ , it is likely that  $\text{Maj}_m(B_i \upharpoonright_\rho)$  is constant.

Since one  $p$ -restriction cannot shrink the formula size sufficiently and leave each majority undetermined, we will build such a restriction incrementally instead.

### 3.2 Staged $p$ -Restrictions

**Proof of Theorem 3.1.** Let  $c'$  be a large constant to be defined later. We first deal with the case that  $\mathcal{L}(h) \leq 2c'$ . Then, since for non-constant  $h$ ,  $\text{Maj}_m$  (or its negation) is a sub-function of  $h \circ \text{Maj}_m$  and since  $\mathcal{L}(\text{Maj}_m) \geq \Omega(m^2)$  ([10]) we get

$$\mathcal{L}(h \circ \text{Maj}_m) \geq \mathcal{L}(\text{Maj}_m) \geq \Omega(m^2) \geq \Omega(\mathcal{L}(h) \cdot m^2)$$

which completes the proof in this case. In the following, we shall assume that  $\mathcal{L}(h) > 2c'$ .

We define the following procedure that runs in  $t$  stages: in the  $j$ -th stage, we generate a  $p_j$ -restriction  $\rho_j$  such that, with high probability, the formula has enough unrestricted variables to balance the number of 0's and 1's and leave enough variables unrestricted for stage  $j + 1$ .

#### Setting Up Parameters

We set

$$m_1 = m$$

and

$$m_{j+1} = m_j^{0.6}$$

for  $j \geq 1$  as long as  $m_j \geq \log^5(4b)$ . Let  $t$  be the last  $j$  such that  $m_j \geq \log^5(4b)$ . A small calculation shows that  $t \leq 2 \log \log m$ . For  $j = 1, \dots, t$  we set

$$p_j = 4m_j^{-0.4} = 4m_{j+1}/m_j$$

#### Shrinkage In $t$ Stages

Denote by  $\varphi_1 = h \circ \text{Maj}_m$ . For  $j = 1, \dots, t$ , we show how to construct  $\varphi_{j+1}$  over variables  $S_{j+1}$  from  $\varphi_j$  over  $S_j$ . We show by induction that  $\varphi_{j+1} = h \circ \text{Maj}_{m_{j+1}}$  (up to a renaming of the variables) and that

$$\mathcal{L}(\varphi_{j+1}) \leq c \cdot \left( \frac{m_{j+1}}{m_j} \right)^2 \cdot \mathcal{L}(\varphi_j),$$

for some large enough universal constant  $c > 0$ .

Let  $j \in \{1, \dots, t\}$ . Let  $\rho_j \in \mathcal{R}_{p_j}$  be a random  $p_j$ -restriction over  $S_j$ . Let

$$X_i^j = \sum_{k=1}^{m_j} X_{ik}^j \qquad Y_i^j = \sum_{k=1}^{m_j} Y_{ik}^j$$

for  $i = 1, \dots, b$  be defined analogously as in the previous section for block  $B_i$ . Then,

$$\mathbb{E}[X_i^j] = m_j p_j = 4m_{j+1}$$

Let  $E_{j,i}$  denote the event that  $|X_i^j - m_j p_j| \leq \frac{1}{4} m_j p_j$  and let  $F_{j,i}$  denote the event that  $|Y_i^j| \leq \frac{1}{2} m_j p_j$ . By Theorem 2.3, using the assumption  $m_j \geq \log^5(4b)$ ,

$$\Pr_{\rho_j \in \mathcal{R}_{p_j}} (E_{j,i}) \geq 1 - 2e^{-2 \frac{m_j^2}{m_j}} = 1 - 2e^{-2m_j^{0.2}} \geq 1 - \frac{1}{4b} \quad (4)$$

$$\Pr_{\rho_j \in \mathcal{R}_{p_j}} (F_{j,i}) \geq 1 - 2e^{-2 \frac{(2m_j+1)^2}{4m_j}} = 1 - 2e^{-2m_j^{0.2}} \geq 1 - \frac{1}{4b} \quad (5)$$

By Corollary 2.2, there exists some constant  $c' > 0$  such that

$$\Pr_{\rho_j \in \mathcal{R}_{p_j}} \left( \mathcal{L}(\varphi_j \upharpoonright_{\rho_j}) \leq c'(1 + p_j^2 \mathcal{L}(\varphi_j)) \right) \geq 0.9$$

Let  $H_j$  denote the event that  $\mathcal{L}(\varphi_j \upharpoonright_{\rho_j}) \leq c' \cdot (1 + p_j^2 \cdot \mathcal{L}(\varphi_j))$ . Thus  $\Pr[H_j] \geq 0.9$ . By the union bound, there exists a restriction  $\rho_j$  for which  $H_j$  and all  $E_{j,i}, F_{j,i}$  for  $i = 1, \dots, b$  hold simultaneously. Fix such a restriction  $\rho_j$ . Now, since  $E_{j,i}$  holds, then

$$X_i^j \geq \frac{3}{4} m_j p_j$$

Since  $F_{j,i}$  also holds, then we can make the number of 0's and 1's equal by fixing at most  $\frac{1}{2} m_j p_j$  variables appropriately, leaving at least  $\frac{1}{4} m_j p_j = m_{j+1}$  unrestricted variables in the block. We restrict the remaining variables further to leave exactly  $m_{j+1}$  unrestricted variables by assigning an equal number of them 0 and 1 in some arbitrary process. Take  $\varphi_{j+1}$  to be the restricted function.

Since  $H_j$  holds, we get

$$\mathcal{L}(\varphi_{j+1}) \leq \mathcal{L}(\varphi_j \upharpoonright_{\rho_j}) \leq c' \cdot (1 + p_j^2 \cdot \mathcal{L}(\varphi_j))$$

However, since  $h$  is a sub-function of  $\varphi_{j+1}$  and since we assumed that  $\mathcal{L}(h) > 2c'$ , we get that  $c' < \frac{1}{2} \mathcal{L}(\varphi_{j+1})$ . Thus,

$$\mathcal{L}(\varphi_{j+1}) < \frac{1}{2} \mathcal{L}(\varphi_{j+1}) + c' \cdot p_j^2 \cdot \mathcal{L}(\varphi_j)$$

which implies that  $\mathcal{L}(\varphi_{j+1}) < 2c' \cdot p_j^2 \cdot \mathcal{L}(\varphi_j)$  and we get

$$\mathcal{L}(\varphi_{j+1}) \leq 2c' \cdot \left( \frac{4m_{j+1}}{m_j} \right)^2 \cdot \mathcal{L}(\varphi_j) = c \cdot \left( \frac{m_{j+1}}{m_j} \right)^2 \cdot \mathcal{L}(\varphi_j)$$

for any  $j \in \{1, \dots, t\}$  by setting  $c = 2c' \cdot 16$ . Overall, we get

$$\mathcal{L}(\varphi_{t+1}) \leq c^t \cdot \left( \frac{m_{t+1}}{m} \right)^2 \cdot \mathcal{L}(\varphi)$$

Since  $h$  is a sub-function of  $\varphi_{t+1}$ , the formula size of  $\varphi_{t+1}$  is at least  $\mathcal{L}(h)$ , which gives

$$\mathcal{L}(\varphi) \geq c^{-t} \cdot \left( \frac{m}{m_{t+1}} \right)^2 \cdot \mathcal{L}(h)$$

Using  $m_{t+1} < \log^5(4b)$  and  $t \leq 2 \log \log m$  we get

$$\mathcal{L}(\varphi) \geq c^{-2 \log \log m} \cdot \left( \frac{m}{\log^5(4b)} \right)^2 \cdot \mathcal{L}(h) \geq \mathcal{L}(h) \cdot m^2 / \text{polylog}(b \cdot m) \quad \blacktriangleleft$$

In the above proof, we only use two facts about the majority function. First, we use that the values of the  $m$ -bit majority function are 0 on inputs  $\vec{x}$  with Hamming weight  $\text{wt}(\vec{x}) = \lceil \frac{m}{2} \rceil - 1$  and 1 on inputs with  $\text{wt}(\vec{x}) = \lceil \frac{m}{2} \rceil$ . In addition, (at the beginning of our proof) we use that  $\mathcal{L}(\text{Maj}_m) \geq \Omega(m^2)$  [10]. Thus our argument extends to any function with these two properties. It turns out that the first condition we need implies the second. Let  $g_m \in \mathcal{B}_m$  be any function such that  $g_m(\vec{x}) = 0$  when  $\text{wt}(\vec{x}) = \lceil \frac{m}{2} \rceil - 1$ , and  $g_m(\vec{x}) = 1$  when  $\text{wt}(\vec{x}) = \lceil \frac{m}{2} \rceil$ . Then by Khrapchenko's theorem [10] the De Morgan formula size of  $g_m$  is at least  $\Omega(m^2)$ .

One can also think of such functions as a partial function that generalizes both Majority and Parity. We obtain the following.

► **Theorem 3.2.** *Let  $m = \frac{n}{\log n}$  and let  $g_m \in \mathcal{B}_m$  be any function such that  $g_m(\vec{x}) = 0$  when  $\text{wt}(\vec{x}) = \lceil \frac{m}{2} \rceil - 1$ , and  $g_m(\vec{x}) = 1$  when  $\text{wt}(\vec{x}) = \lceil \frac{m}{2} \rceil$ . Then,*

$$\mathcal{L}(\mathcal{A}_n^{g_m}) \geq n^3 / \text{polylog}(n)$$

## 4 Consequences

### 4.1 Composition with Other Threshold Functions

We also obtain lower bounds for compositions with arbitrary threshold functions  $\text{Th}_{m,k}$  instead of  $\text{Maj}_m$ . We use that  $\text{Th}_{2k+1,k}$  is a subfunction of  $\text{Th}_{m,k}$ . Fixing arbitrary  $m - (2k + 1)$  bits to 0 in each block, our results immediately imply that  $\mathcal{L}(h \circ \text{Th}_{m,k}) \geq \mathcal{L}(h) \cdot k^2 / \text{polylog}(k, b)$ . We get stronger bounds by noticing that fixing the  $m - (2k + 1)$  heaviest variables in each block, the formula shrinks by a factor of  $(2k + 1)/m$ . Thus, we get

$$\mathcal{L}(h \circ \text{Th}_{m,k}) \geq \mathcal{L}(h) \cdot m \cdot k / \text{polylog}(k, b)$$

This implies the following:

► **Theorem 4.1.** *Let  $m = \frac{n}{\log n}$  and  $k \leq m/2$ . Then*

$$\mathcal{L}(\mathcal{A}_n^{\text{Th}_{m,k}}) \geq n^2 \cdot k / \text{polylog}(n)$$

### 4.2 Iterated Compositions

Since the composed function “Parity of Parities” is just Parity, considering iterated compositions in place of Parity in the original lower bound arguments for Andreev function did not give new functions. But taking iterated compositions of Majorities yield new functions, such as “Majority of Majorities”, “Parity of Majorities”, “Majority of Parities” and so on. Our results extend to the generalized Andreev function with iterated compositions in place of  $g_m$ . We obtain additional functions with cubic formula size lower bounds.

► **Theorem 4.2.** *Let  $\mathcal{G}_m$  denote the set of functions  $g_m \in \mathcal{B}_m$  such that  $g_m(\vec{x}) = 0$  when  $\text{wt}(\vec{x}) = \lceil \frac{m}{2} \rceil - 1$ , and  $g_m(\vec{x}) = 1$  when  $\text{wt}(\vec{x}) = \lceil \frac{m}{2} \rceil$ ; or the other way around, that is  $g_m(\vec{x}) = 1$  when  $\text{wt}(\vec{x}) = \lceil \frac{m}{2} \rceil - 1$ , and  $g_m(\vec{x}) = 0$  when  $\text{wt}(\vec{x}) = \lceil \frac{m}{2} \rceil$ .*

*Let  $m = \frac{n}{\log n}$  and let  $u \geq 2$  and  $v \geq 2$  be integers such that  $uv = m$ . For any functions  $f_u \in \mathcal{G}_u$  and  $g_v \in \mathcal{G}_v$*

$$\mathcal{L}(\mathcal{A}_n^{f_u \circ g_v}) \geq n^3 / \text{polylog}(n)$$



**Proof.** Let  $h \in \mathcal{B}_{\log n}$  be a function of maximum formula complexity. By Theorem 3.1

$$\mathcal{L}(h \circ f_u) \geq \mathcal{L}(h) \cdot u^2 / \text{polylog}(b \cdot u)$$

where  $b = \log n$ . Let  $b' = b \cdot u$ . By Theorem 3.1

$$\mathcal{L}(h \circ f_u \circ g_v) \geq \mathcal{L}(h \circ f_u) \cdot v^2 / \text{polylog}(b' \cdot v)$$

Thus,

$$\mathcal{L}(h \circ f_u \circ g_v) \geq \mathcal{L}(h) \cdot \frac{u^2}{\text{polylog}(b \cdot u)} \cdot \frac{v^2}{\text{polylog}(b \cdot u \cdot v)} \geq \mathcal{L}(h) \cdot \frac{m^2}{\text{polylog}(n)} \geq \frac{n^3}{\text{polylog}(n)} \quad \blacktriangleleft$$

The argument extends to repeated iterations. As the proof shows, we lose a  $\text{polylog}(n)$  factor from the  $n^3$  lower bound at each iteration.

### 4.3 Cubic Formula Size Lower Bounds for an Explicit Monotone Function

A function  $h : \{0, 1\}^b \rightarrow \{0, 1\}$  is called a *slice function* if on inputs  $\vec{z} \in \{0, 1\}^b$ ,  $h(\vec{z}) = 1$  if  $\text{wt}(\vec{z}) \geq \lfloor \frac{b}{2} \rfloor + 1$ , and  $h(\vec{z}) = 0$  if  $\text{wt}(\vec{z}) < \lfloor \frac{b}{2} \rfloor$ . Note that every slice function is monotone, and slice functions differ from each other only on inputs in the middle layer of the Boolean cube, that is on inputs with weight exactly  $\lfloor \frac{b}{2} \rfloor$ .

The *monotone address function* defined by Wegener [21] takes  $b + n$  input bits where  $n = \binom{b}{\lfloor \frac{b}{2} \rfloor}$ . The  $n$  bits are interpreted to specify a slice function  $h$  on  $b$  bits. We denote by  $h$  both the  $n$ -bit string and the slice function specified by it. Then, on input  $(z, h)$  where  $z \in \{0, 1\}^b$  and  $h \in \{0, 1\}^n$ , the output of the monotone address function is  $h(z)$ . Note that the monotone address function itself is monotone.

We are now ready to define a monotone function that requires cubic formula size. Let  $n = \binom{b}{\lfloor b/2 \rfloor}$ , and let  $m = n/b$ . Similarly to the Generalized Andreev Function, we define a function  $\mathcal{F}_n : \{0, 1\}^n \times \{0, 1\}^n \rightarrow \{0, 1\}$  on  $2n$  bits.

The function  $\mathcal{F}_n$  takes two inputs: an  $n$ -bit string representing a slice function  $h$  on  $b$  bits, and an  $n$ -bit string  $\vec{x}$ , interpreted as a  $b \times m$  matrix with rows  $\vec{x}_1, \dots, \vec{x}_b$ . Then,

$$\mathcal{F}_n(h, \vec{x}) = (h \circ \text{Maj}_m)(\vec{x}) = h(\text{Maj}_m(\vec{x}_1), \dots, \text{Maj}_m(\vec{x}_b))$$

We can further generalize this as follows: If  $g_m \in \mathcal{B}_m$  is an arbitrary function on  $m$  bits, then let  $\mathcal{F}_n^{g_m} : \{0, 1\}^n \times \{0, 1\}^n \rightarrow \{0, 1\}$  denote the function on  $2n$  bits, defined analogously by

$$\mathcal{F}_n^{g_m}(h, \vec{x}) = (h \circ g_m)(\vec{x}) = h(g_m(\vec{x}_1), \dots, g_m(\vec{x}_b))$$

In particular,  $\mathcal{F}_n = \mathcal{F}_n^{\text{Maj}_m}$ . Note that for any monotone function  $g_m$ , the function  $\mathcal{F}_n^{g_m}$  is also monotone.

Since the number of De Morgan formulas of size  $s$  on  $b$  input bits is at most  $(cb)^s$  for some constant  $c$  [15], and the number of different slice functions on  $b$  input bits is  $2^n$  where  $n = \binom{b}{\lfloor b/2 \rfloor}$ , by a standard counting argument, there are slice functions on  $b$  bits that require formula size at least  $\Omega(\frac{n}{\log b}) = \Omega(\frac{n}{\log \log n})$ .

This implies the following bound on the formula size of the monotone function  $\mathcal{F}_n$ .

► **Theorem 4.3.**

$$\mathcal{L}(\mathcal{F}_n) \geq n^3 / \text{polylog}(n)$$

#### 4.4 Monotone Formula Size of Majority

Our results highlight again the question of determining the shrinkage exponent for monotone formulas, raised by Håstad [5]. It was pointed out by Håstad [5], that determining the shrinkage exponent for monotone formulas could potentially yield improved lower bounds on the monotone formula size of the Majority function. Our results make this connection explicit, without any dependence on how the value of the shrinkage exponent is obtained. More precisely, our arguments imply the following.

► **Theorem 4.4.** *Let  $\Gamma_{\text{mon}}$  denote the shrinkage exponent of monotone formulas. Then  $\mathcal{L}_{\text{mon}}(\text{Maj}_n) \geq n^{\Gamma_{\text{mon}}} / \text{polylog } n$ , where  $\mathcal{L}_{\text{mon}}$  denotes monotone formula complexity.*

**Proof.** To see this, notice that our argument in the proof of Theorem 3.1 can also be carried out when  $b = 1$  and  $h : \{0, 1\}^1 \rightarrow \{0, 1\}$  is the identity function, that is  $h \circ \text{Maj}_m = \text{Maj}_m$ , and we apply our staged restrictions on just one block.

Let  $\Gamma = \Gamma_{\text{mon}}$ . Then by definition,

$$\mathbb{E}_{\rho \sim \mathcal{R}_p} \left[ \mathcal{L}_{\text{mon}} \left( f \upharpoonright_{\rho} \right) \right] = \mathcal{O} \left( 1 + p^{\Gamma} \mathcal{L}_{\text{mon}}(f) \right)$$

Let  $c'$  be a constant such that

$$\mathbb{E}_{\rho \sim \mathcal{R}_p} \left[ \mathcal{L}_{\text{mon}} \left( f \upharpoonright_{\rho} \right) \right] \leq c' \cdot \left( 1 + p^{\Gamma} \mathcal{L}_{\text{mon}}(f) \right)$$

Let  $m_1 = m = n$ . As in the proof of Theorem 3.1, we set  $m_{j+1} = m_j^{0.6}$  for  $j \geq 1$ . Let  $t$  be the last  $j$  such that  $m_j \geq 32$  and  $\mathcal{L}_{\text{mon}}(\text{Maj}_{m_{j+1}}) \geq 2c'$  both hold. (Recall that  $b = 1$ , thus  $\log^5(4b) = 2^5 = 32$ .)

A small calculation shows that  $t \leq \frac{1}{\log(10/6)} \log \log m \leq 2 \log \log m$ . For  $j = 1, \dots, t$  we set  $p_j = 4m_j^{-0.4} = 4m_{j+1}/m_j$ .

As in the proof of Theorem 3.1, our staged restrictions ensure that  $\varphi_j = \text{Maj}_{m_j}$ . Similarly to our previous argument, setting  $c = 2c' \cdot 16$ , and using that

$$\mathcal{L}_{\text{mon}}(\varphi_{j+1}) = \mathcal{L}_{\text{mon}}(\text{Maj}_{m_{j+1}}) \geq 2c'$$

for  $j = 1, \dots, t$ , we get

$$\mathcal{L}_{\text{mon}}(\varphi_{t+1}) \leq c^t \cdot \left( \frac{m_{t+1}}{m} \right)^{\Gamma} \cdot \mathcal{L}_{\text{mon}}(\varphi_1)$$

Thus, we obtain

$$\mathcal{L}_{\text{mon}}(\text{Maj}_n) \geq c^{-t} \cdot \left( \frac{n}{m_{t+1}} \right)^{\Gamma} \cdot \mathcal{L}_{\text{mon}}(\text{Maj}_{m_{t+1}})$$

By the definition of  $t$  above, at least one of  $m_{t+1} < 32$  or  $\mathcal{L}_{\text{mon}}(\text{Maj}_{m_{t+2}}) < 2c'$  must hold. The latter implies that  $m_{t+2} < 2c'$ , hence  $m_{t+1} = m_{t+2}^{10/6} < (2c')^{10/6}$  and for  $c'' = \max\{32, (2c')^{10/6}\}$  we have  $m_{t+1} < c''$ . Since  $m_t \geq 32$  we also have  $m_{t+1} \geq 8$ . Using  $8 \leq m_{t+1} < c''$  and  $t \leq 2 \log \log m$  we get

$$\mathcal{L}_{\text{mon}}(\text{Maj}_n) \geq c^{-2 \log \log m} \cdot \left( \frac{n}{c''} \right)^{\Gamma} \cdot \mathcal{L}_{\text{mon}}(\text{Maj}_{m_{t+1}}) \geq n^{\Gamma} / \text{polylog } n \quad \blacktriangleleft$$

## 5 Future Directions

A possible extension of our result would be to verify the KRW conjecture [9] for composing arbitrary functions with the majority function. The KRW conjecture essentially states that the formula size of composed functions is the product of their formula sizes, e.g.  $\mathcal{L}(h \circ g) \geq \Omega(\mathcal{L}(h) \cdot \mathcal{L}(g))$ . The conjecture has been verified for composing arbitrary functions with parity. Unfortunately, getting asymptotically tight bounds on the formula size of majority is still open. Currently, the best upper bound on the De Morgan formula size of majority is  $\mathcal{O}(n^{3.91})$  [16]. Our lower bound would verify the conjecture for composing arbitrary functions with majority if  $\mathcal{L}(\text{Maj}_n) = \mathcal{O}(n^2)$ .

Another interesting direction is studying the average-case hardness of the Generalized Andreev function with Majority. Here, we expect a different behavior than the standard Andreev function that is hard to compute on  $1/2 + \exp(-n^{\Omega(1)})$  fraction of the inputs [11] (under the uniform distribution). For  $\mathcal{M}_n$  we could not hope to get such strong average-case hardness, as we argue next. Observe that a Majority function on the  $\{x_1, \dots, x_m\}$  agrees with the dictator function of  $x_1$  on  $1/2 + \Omega(1/\sqrt{m})$  fraction of the inputs. Replacing each majority in  $\mathcal{M}_n$  with the appropriate dictator yields the address function, which has formula complexity  $\Theta(n)$ . A small calculation shows that that a linear size formula (computing the address function) has agreement at least  $1/2 + \Omega(1/\sqrt{m})^{\log n} \geq 1/2 + 2^{-\log^2(n)}$  with  $\mathcal{M}_n$ . We conjecture that getting a much better agreement with  $\mathcal{M}_n$ , say  $1/2 + 1/\text{poly}(n)$ , or even  $1/2 + 2^{-o(\log^2 n)}$ , requires almost cubic formula complexity.

---

## References

- 1 A. E. Andreev. On a method for obtaining more than quadratic effective lower bounds for the complexity of  $\pi$ -scheme. *Moscow University Mathematics Bulletin*, 42(1):63–66, 1987.
- 2 Andrej Bogdanov. Small Bias Requires Large Formulas. In *45th International Colloquium on Automata, Languages, and Programming, ICALP 2018, July 9-13, 2018, Prague, Czech Republic*, pages 22:1–22:12, 2018. doi:10.4230/LIPIcs.ICALP.2018.22.
- 3 Ruiwen Chen, Valentine Kabanets, Antonina Kolokolova, Ronen Shaltiel, and David Zuckerman. Mining Circuit Lower Bound Proofs for Meta-Algorithms. *Computational Complexity*, 24(2):333–392, June 2015. doi:10.1007/s00037-015-0100-0.
- 4 Irit Dinur and Or Meir. Toward the KRW Composition Conjecture: Cubic Formula Lower Bounds via Communication Complexity. In *31st Conference on Computational Complexity, CCC 2016, May 29 to June 1, 2016, Tokyo, Japan*, pages 3:1–3:51, 2016. doi:10.4230/LIPIcs.CCC.2016.3.
- 5 Johan Håstad. The Shrinkage Exponent of de Morgan Formulas is 2. *SIAM Journal on Computing*, 27(1):48–64, 1998. doi:10.1137/S0097539794261556.
- 6 Wassily Hoeffding. Probability inequalities for sums of bounded random variables. *Journal of the American statistical association*, 58(301):13–30, 1963.
- 7 Russell Impagliazzo, Raghu Meka, and David Zuckerman. Pseudorandomness from Shrinkage. In *53rd Annual IEEE Symposium on Foundations of Computer Science, FOCS 2012, New Brunswick, NJ, USA, October 20-23, 2012*, pages 111–119, 2012. doi:10.1109/FOCS.2012.78.
- 8 Russell Impagliazzo and Noam Nisan. The effect of random restrictions on formula size. *Random Structures & Algorithms*, 4(2):121–133, 1993.
- 9 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.

- 10 V. M. Khrapchenko. Complexity of the realization of a linear function in the class of  $\Pi$ -circuits. *Mathematical notes of the Academy of Sciences of the USSR*, 9(1):21–23, January 1971. doi:10.1007/BF01405045.
- 11 Ilan Komargodski and Ran Raz. Average-case lower bounds for formula size. In *Symposium on Theory of Computing Conference, STOC'13, Palo Alto, CA, USA, June 1-4, 2013*, pages 171–180, 2013. doi:10.1145/2488608.2488630.
- 12 Ilan Komargodski, Ran Raz, and Avishay Tal. Improved Average-Case Lower Bounds for De Morgan Formula Size: Matching Worst-Case Lower Bound. *SIAM Journal on Computing*, 46(1):37–57, 2017. doi:10.1137/15M1048045.
- 13 Michael S Paterson and Uri Zwick. Shrinkage of de Morgan formulae under restriction. *Random Structures & Algorithms*, 4(2):135–150, 1993.
- 14 P. Pudlak. Personal communication, 2018.
- 15 John Riordan and Claude E Shannon. The Number of Two-Terminal Series-Parallel Networks. *Studies in Applied Mathematics*, 21(1-4):83–93, 1942.
- 16 I. S. Sergeev. Complexity and depth of formulas for symmetric Boolean functions. *Moscow University Mathematics Bulletin*, 71(3):127–130, 2016.
- 17 Bella Abramovna Subbotovskaya. Realizations of linear functions by formulas using  $+$ ,  $*$ , and  $-$ . *Doklady Akademii Nauk SSSR*, 136(3):553–555, 1961.
- 18 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. doi:10.1109/FOCS.2014.65.
- 19 Avishay Tal. Formula lower bounds via the quantum method. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2017, Montreal, QC, Canada, June 19-23, 2017*, pages 1256–1268, 2017. doi:10.1145/3055399.3055472.
- 20 Leslie G. Valiant. Short monotone formulae for the majority function. *Journal of Algorithms*, 5(3):363–366, 1984.
- 21 I. Wegener. The critical complexity of all (monotone) Boolean functions and monotone graph properties. *Information and Control*, 67:212–222, 1985.



# The Space Complexity of Mirror Games

**Sumegha Garg**

Princeton University, Princeton, USA  
sumeghag@cs.princeton.edu

**Jon Schneider**

Google Research, New York, USA  
jschnei@google.com

---

## Abstract

We consider the following game between two players Alice and Bob, which we call the mirror game. Alice and Bob take turns saying numbers belonging to the set  $\{1, 2, \dots, N\}$ . A player loses if they repeat a number that has already been said. Otherwise, after  $N$  turns, when all the numbers have been spoken, both players win. When  $N$  is even, Bob, who goes second, has a very simple (and memoryless) strategy to avoid losing: whenever Alice says  $x$ , respond with  $N + 1 - x$ . The question is: does Alice have a similarly simple strategy to win that avoids remembering all the numbers said by Bob?

The answer is no. We prove a linear lower bound on the space complexity of any deterministic winning strategy of Alice. Interestingly, this follows as a consequence of the Eventown-Oddtown theorem from extremal combinatorics. We additionally demonstrate a randomized strategy for Alice that wins with high probability that requires only  $\tilde{O}(\sqrt{N})$  space (provided that Alice has access to a random matching on  $K_N$ ).

We also investigate lower bounds for a generalized mirror game where Alice and Bob alternate saying 1 number and  $b$  numbers each turn (respectively). When  $1 + b$  is a prime, our linear lower bounds continue to hold, but when  $1 + b$  is composite, we show that the existence of a  $o(N)$  space strategy for Bob (when  $N \not\equiv 0 \pmod{1 + b}$ ) implies the existence of exponential-sized matching vector families over  $\mathbb{Z}_{1+b}^N$ .

**2012 ACM Subject Classification** Theory of computation → Interactive computation

**Keywords and phrases** Mirror Games, Space Complexity, Eventown-Oddtown

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2019.36

**Related Version** <https://arxiv.org/abs/1710.02898>

**Acknowledgements** We would like to thank Mark Braverman, Sivakanth Gopi, and Jieming Mao for helpful advice and discussions.

## 1 Introduction

### 1.1 The Mirror Game and Mirror Strategies

Consider the following simple game. Alice and Bob take turns saying numbers belonging to the set  $\{1, 2, \dots, N\}$ . If either player says a number that has previously been said, they lose. Otherwise, after  $N$  turns, all the numbers in the set have been spoken aloud, and both players win. Alice says the first number.

If  $N$  is even, there is a very simple and computationally efficient strategy that allows Bob to win this game, regardless of Alice's strategy: whenever Alice says  $x$ , Bob replies with  $N + 1 - x$ . This is an example of a *mirror strategy* (and for this reason, we refer to the game above as the *mirror game*). Mirror strategies are an effective tool for figuring out who wins



© Sumegha Garg and Jon Schneider;  
licensed under Creative Commons License CC-BY  
10th Innovations in Theoretical Computer Science (ITCS 2019).  
Editor: Avrim Blum; Article No. 36; pp. 36:1–36:14



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

in a variety of combinatorial games (for example, two-pile Nim [2]). More practically, mirror strategies can be applied when playing more complex games, such as chess or go (to varying degrees of success). From a computational perspective, mirror strategies are interesting as they require very limited computational resources - most mirror strategies can be described via a simple transformation of the preceding action.

Returning to the game above, this leads to the following natural question: does Alice have a simple strategy to avoid losing when  $N$  is even? Since both players have access to the same set of actions, one may be tempted to believe that the answer is yes - in fact, if  $N$  is odd, then Alice can start by saying the number  $N$  and then adopt the mirror strategy for Bob described above for a set of  $N - 1$  elements. However, when  $N$  is even, the mirror strategy as stated does not work.

To answer the above question, we need to formalize what we mean by simple; after all, Alice has plenty of strategies such as “say the smallest number which has not yet been said”. One useful metric of simplicity is the metric of *space complexity*, the amount of memory a player requires to implement their strategy (we formalize this in Section 2). Note that Bob needs only  $O(\log N)$  bits of memory to implement his mirror strategy, whereas the naive strategy for Alice (remembering everything) requires  $O(N)$  bits.

In this paper, we show that this gap is necessary; any successful, deterministic strategy for Alice requires at least  $\Omega(N)$  bits of memory:

► **Theorem 1** (Restatement of Theorem 11). *If  $N$  is even, then any winning strategy for Alice in the mirror game requires at least  $(\log_2 5 - 2)N - o(N)$  bits of space.*

## 1.2 Eventown and Oddtown

While many tools exist in the computer science literature for showing space lower bounds (e.g. communication complexity, information theory, etc.), one interesting feature of this problem absent from many others is that any proof of Theorem 1 *must depend crucially on the parity* of  $N$ .

In the study of set families in extremal combinatorics, an “Oddtown” is a collection of subsets of  $\{1, 2, \dots, N\}$  where every subset has even cardinality but each pair of distinct subsets has an intersection of odd cardinality. Likewise, an “Eventown” is a collection of subsets of  $\{1, 2, \dots, N\}$  where every subset has even cardinality but each pair of distinct subsets has an intersection of even cardinality. In 1969, Berlekamp [3], answering a question of Erdős, showed that while there exist Eventowns containing up to  $2^{N/2}$  subsets, any Oddtown contains at most  $N$  subsets.

It turns out that this exponential gap between the size of Oddtowns and the size of Eventowns is directly responsible for the exponential gap between the space complexity of Alice’s strategy and the space complexity of Bob’s strategy! (One way to see this connection is that, just as Bob’s  $O(\log N)$ -space strategy involves pairing up the numbers of  $\{1, 2, \dots, N\}$ , one way to construct an Eventown of size  $2^{N/2}$  is to perform a similar pairing, and then consider all subsets formed by unions of pairs).

The Eventown-Oddtown theorem figures into our proof of Theorem 1 in the following way. At a given turn, for each possible state of memory of Alice, label the state with the possible subsets of numbers that could have possibly been said before this turn. We show (Lemma 13) that these subsets must form an Oddtown, or else Bob has some strategy that can force Alice to lose. Since there are a large (exponential) number of total possible subsets (Corollary 10), and since each Oddtown contains at most  $N$  subsets, this implies that Alice’s memory must be large.



### 1.3 Randomized Strategies for Alice

A natural followup to Theorem 1 is whether these lower bounds continue to hold if Alice instead uses a randomized strategy, which only needs to succeed with high probability.

We provide some evidence to show that this might not be the case. We demonstrate an  $O(\sqrt{N} \log^2 N)$ -space algorithm for Alice that succeeds with probability  $1 - O(1/N)$ , as long as Alice is provided access to a uniformly chosen perfect matching on the complete graph on  $N$  vertices ( $K_N$ ) (Theorem 18). In addition, even with  $O(\log N)$ -space (and access to a uniformly chosen perfect matching on  $K_N$ ), Alice can guarantee success with probability  $\Omega(1/N)$ .

In both of these algorithms, Alice attempts the mirror strategy of Bob, hoping that Bob does not choose the number with no match. In the  $O(\sqrt{N} \log^2 N)$  algorithm, Alice decreases her probability of failure by maintaining a set of  $\tilde{O}(\sqrt{N})$  possible “backup” points she can switch to if Bob identifies the unmatched point. This lets Alice survive until turn  $N - \tilde{O}(\sqrt{N})$ , whereupon Alice can reconstruct the remaining  $O(\sqrt{N})$  elements by maintaining  $O(\sqrt{N})$  power sums during the computation.

Since Alice cannot store a perfect matching on  $K_N$  in  $o(N)$  space, this unfortunately does not give any  $o(N)$ -space strategies for Alice. In addition, our lower bound techniques from Theorem 1 fail to give non-trivial guarantees in the randomized model. Demonstrating non-trivial algorithms or non-trivial lower bounds for the general case is an interesting open problem.

### 1.4 General Mirror Games

It is possible to generalize the mirror game presented earlier to the case where in each turn Alice says  $a$  numbers and Bob says  $b$  numbers. We refer to this game as the  $(a, b)$ -mirror game.

By a generalization of the Oddtown theorem that works modulo prime  $p$ , our proof of Theorem 1 immediately carries over to show that if  $N$  is not divisible by  $p$ , then any winning strategy for Bob in the  $(1, p - 1)$ -mirror game requires  $\Omega(N)$  space (Theorem 20).

The natural generalization of the Oddtown theorem is known not to hold modulo composite  $m$ . Grolmusz showed [9] that if  $m$  is composite, there exists a set family of quasipolynomial size such that every set has cardinality divisible by  $m$ , but the intersection of any two distinct sets has cardinality not divisible by  $m$ . The best known upper bounds for the size of such a set family are of the form  $2^{O(N)}$ , and improving these to any bound of the form  $2^{o(N)}$  is an important open problem, with applications in coding theory to the construction of matching vector families [5, 4].

As long as the size of a modulo  $m$  Oddtown is bounded above by  $2^{o(N)}$ , the proof of Theorem 20 still achieves an  $\Omega(N)$  lower bound on the space complexity of Bob’s strategy. It follows that finding any  $o(N)$ -space winning strategy for Bob in the  $(1, m - 1)$  mirror game implies the existence of a  $2^{\Omega(N)}$ -size modulo  $m$  Oddtown, and hence similarly sized matching vector families (Theorem 22). Admittedly, Bob may not have an  $o(N)$ -space winning strategy in this game for unrelated reasons; it is interesting whether it is possible to show a converse result, constructing a low-space strategy for Bob from any  $2^{\Omega(N)}$ -size modulo  $m$  Oddtown.

For other  $(a, b)$ -mirror games, we understand much less about the complexity of winning strategies (if  $a + b$  is prime, then depending on the residue of  $N$  modulo  $a + b$  it is occasionally possible to show an  $\Omega(N)$  lower bound on the lower bound of a winning strategy).

## 2 Definitions and Preliminaries

### 2.1 The Mirror Game

The *mirror game on  $N$  elements* as discussed before is a game between two players, Alice and Bob. Alice and Bob take turns (Alice going first) saying a number in  $[N]$  ( $[N]$  indicates the set  $\{1, 2, \dots, N\}$ ). If a player says a number that has previously been said (either by the other player or by themselves) they lose and the other player wins. In addition, after  $N$  successful turns (so no numbers in  $[N]$  remain unsaid), both players are declared winners.

With unbounded memory, it is clear that both players can easily win this game. We therefore restrict our attention to strategies with bounded memory. A *strategy computable in memory  $m$*  for Alice in the mirror game is defined by an initial memory state  $s_0 \in [2^m]$  and a pair of transition functions  $f : [N] \times [2^m] \times [N] \rightarrow [2^m]$  and  $g : [2^m] \rightarrow [N]$  computable in  $SPACE(m)$  (see [1] for an introduction to bounded space complexity). The function  $f$  takes in the previous reply  $b \in [N]$  of Bob, Alice's current memory state  $s \in [2^m]$ , and the current turn  $t \in [N]$ , and returns Alice's new memory state  $s' \in [2^m]$ ; the function  $g$  takes in Alice's current memory state  $s \in [2^m]$  (after updating it based on Bob's move), and outputs her next move  $a$ . We say a strategy for Alice is a *winning strategy* if Alice is guaranteed to win regardless of Bob's choice of actions.

By removing the constraint that the transition function  $f$  is computable in  $SPACE(m)$ , we obtain a larger class of strategies, which we refer to as strategies *weakly computable in memory  $m$* . Our lower bounds in Theorems 1 and 20 continue to hold for this larger class of strategies.

### 2.2 Eventown and Oddtown

In this section we review the known literature on the Eventown-Oddtown problem. Note that while in the introduction we only defined the terms "Eventown" and "Oddtown", there are actually four different classes of set system depending on the parity of cardinalities of the subsets and the parity of the cardinality of the intersection.

► **Definition 2.** A collection of subsets  $\mathcal{F} \subseteq [N]$  forms an *(Odd, Even)-town* of sets if:

1. For every  $F \in \mathcal{F}$ ,  $|F| \equiv 1 \pmod{2}$ .
2. For every  $F_1 \neq F_2 \in \mathcal{F}$ ,  $|F_1 \cap F_2| = 0 \pmod{2}$ .

We define *(Odd, Odd)-towns*, *(Even, Odd)-towns*, and *(Even, Even)-towns* similarly.

Note that there exist (Even, Even)-towns and (Odd, Odd)-towns containing exponentially (in  $N$ ) many sets; one simple construction of an (Even, Even)-town is to partition the ground set into pairs (possibly with a leftover element), and consider all sets formed by taking unions of these pairs. In contrast, (Even, Odd)-towns and (Odd, Even)-towns each contain at most  $N$  sets.

► **Lemma 3.** Any *(Odd, Even)-town*  $\mathcal{F}$  has size at most  $N$  i.e.  $|\mathcal{F}| \leq N$ . [3, 12]

For completeness, we give the proof below.

**Proof.** Let  $\mathcal{F}$  be  $\{F_1, F_2, \dots, F_k\}$ . Let  $v_i \in \{0, 1\}^N$  be the characteristic function of  $F_i$  i.e.  $v_i[l] = 1 \iff l \in F_i$ . We know that  $v_i^T v_j \pmod{2} = 1$  for  $i = j$  and 0 otherwise. We claim that  $v_1, v_2, \dots, v_k$  are linearly independent over  $\mathbb{F}_2$  (where  $\mathbb{F}_2$  is the field with 2 elements) and hence,  $k \leq N$ . We prove this claim by contradiction. If they are not linearly independent, then  $\exists \lambda_1, \lambda_2, \dots, \lambda_k \in \{0, 1\}$  not all zero such that  $\sum_{i=1}^k \lambda_i v_i = 0 \pmod{2}$ . Without loss of generality, assume  $\lambda_1 \neq 0$ . As  $v_1^T (\sum_{i=1}^k \lambda_i v_i) = \lambda_1$  over  $\mathbb{F}_2$ , this implies  $\lambda_1 = 0$ ; a contradiction. ◀

► **Corollary 4.** *Any (Even, Odd)-town contains at most  $N$  sets.*

**Proof.** We prove an upper bound of  $N + 1$  sets by reduction to Lemma 3. Given an (Even, Odd)-town  $\mathcal{B} = \{B_1, \dots, B_m\}$ , construct an (Odd, Even)-town in space  $[N + 1]$  from sets  $B_1 \cup \{N + 1\}, B_2 \cup \{N + 1\}, \dots, B_m \cup \{N + 1\}$ . Using Lemma 3, we get  $m \leq N + 1$ . For a stronger upper bound of  $N$ , refer to [3, 12]. ◀

By adapting the above linear algebraic arguments to the field  $\mathbb{F}_p$  ( $p$  is prime and  $\mathbb{F}_p$  refers to field over  $p$  elements), it is possible to show similar upper bounds on the size of set families with cardinality constraints modulo  $p$ . We will use the following lemma, due to Frankl and Wilson.

► **Lemma 5 ([8]).** *Let  $p$  be a prime and  $L$  be a set of  $s$  integers. Let  $B_1, B_2, \dots, B_m \subseteq [N]$  be a family of subsets such that:*

1.  $|B_i| \bmod p \notin L, \forall i \in [m]$ .
2.  $|B_i \cap B_j| \bmod p \in L, \forall i \neq j$ .

Then

$$m \leq \sum_{i=0}^s \binom{N}{s}$$

We call such family of subsets a  $(p, L)$ -Modtown.

Interestingly, by a result of Grolmusz, there is no straightforward generalization of these results modulo a composite number  $k$ .

► **Theorem 6.** *Let  $k$  be a positive integer with  $r > 1$  different prime divisors. Then there exists a  $c > 0$ , such that for every  $N$ , there exists a family of subsets  $B_1, B_2, \dots, B_m$  such that:*

1.  $m \geq \exp(c(\log N)^r / (\log \log N)^{r-1})$ ,
2.  $|B_i| \equiv 0 \pmod{k}, \forall i$ ,
3.  $|B_i \cap B_j| \not\equiv 0 \pmod{k}, \forall i \neq j$ .

**Proof.** See [9]. ◀

This type of set family is captured in coding theory by the definition of a matching vector family.

► **Definition 7.** A *matching vector family* [4] over  $\mathbb{Z}_m^n$  of size  $t$  is a pair of ordered lists  $U = \{u_1, u_2, \dots, u_t\}$  and  $V = \{v_1, v_2, \dots, v_t\}$  where  $u_i, v_j \in \mathbb{Z}_m^n$  such that for all  $i$ ,  $u_i^T v_i = 0$ , and for all  $i \neq j$ ,  $u_i^T v_j \neq 0$ .

By taking  $u_i = v_i$  to be the characteristic vector of  $B_i$  above, it is clear that a family of subsets of size  $m$  in Theorem 6 gives rise to a Matching Vector family of size  $m$ . Matching vector families have deep applications to many problems in coding theory, such as private information retrieval and locally decodable codes [6, 7]. Understanding the maximum possible size of a matching vector family is an important open problem in coding theory.

### 3 Alice Requires Linear Space

In this section we prove that Alice requires linear space to win the mirror game when  $N$  is even. To do this, we will show that if Alice is at a specific memory state, then the possible sets of numbers that have been said till that point in time must form an (Even, Odd)-town (and hence there are at most  $N$  such sets for any memory state).

Before we get into the main proof, we will define and lower-bound the size of what we call a “covering collection” of subsets (this will later allow us to lower bound the total number of possible sets of numbers said by round  $t$  ( $2t$  turns)).

► **Definition 8.** A collection of subsets  $\mathcal{C}$  of  $[N]$  is  $(p, r)$ -covering if:

1. each  $S \in \mathcal{C}$  has  $|S| = pr$ .
2. for every  $T \subset [N]$  with  $|T| = r$ , there exists a  $S \in \mathcal{C}$  with  $T \subset S$ .

► **Lemma 9.** Every  $(p, r)$ -covering collection  $\mathcal{C}$  has size at least  $\frac{\binom{N}{r}}{\binom{pr}{r}}$ .

**Proof.** Every set in this collection contains at most  $\binom{pr}{r}$  sets  $T$  with cardinality  $r$ . There are  $\binom{N}{r}$  possible sets  $T$ . ◀

When  $p = 2$ , it turns out that the lower bound in Lemma 9 is maximized when  $r = N/5$ .

► **Corollary 10.** When  $r = N/5$ , every  $(2, r)$ -covering collection has size at least  $2^{(\log_2 5 - 2)N - o(N)}$ .

We now prove our main theorem.

► **Theorem 11.** If  $N$  is even, then any winning strategy in the mirror game for Alice requires at least  $(\log_2 5 - 2)N - o(N)$  bits of space.

**Proof.** Fix a winning strategy for Alice. Assume this strategy uses  $m$  bits of memory, and thus has  $M = 2^m$  distinct states of memory.

Call a subset  $S$  of  $[N]$   $r$ -occurring if it is possible that immediately after turn  $2r$  ( $r$  rounds), the set of numbers that have been said is equal to  $S$ . Let  $\mathcal{S}_r$  be the collection of all  $r$ -occurring sets. Before diving into the main proof, for any fixed deterministic strategy of Alice, we prove a lower bound on  $\mathcal{S}_r$  i.e. the number of different subsets of numbers that could have been said in the first  $2r$  turns over various strategies of Bob.

► **Lemma 12.**  $\mathcal{S}_r$  is  $(2, r)$ -covering.

**Proof.** Since  $2r$  numbers have been said immediately after turn  $2r$ , every set in  $\mathcal{S}_r$  has cardinality  $2r$ . We must show that for any  $T \subset [N]$  with  $|T| = r$ , that there exists an  $S$  in  $\mathcal{S}_r$  with  $T \subset S$ .

Consider the following strategy for Bob: “say the smallest number in  $T$  which has not yet been said”. Note that if Bob follows this strategy, then the set of numbers said by turn  $2r$  must contain the entire set  $T$ . This set belongs to  $\mathcal{S}_r$ , and it follows that  $\mathcal{S}_r$  is  $(2, r)$ -covering. ◀

Write  $N = 2n$ , and fix a value  $r \in [n]$ . For a memory state  $x$  out of the  $M$  possible memory states and an  $r$ -occurring set  $S$ , label  $x$  with  $S$  if it is possible that Alice is at memory state  $x$  when the set of numbers that have been said is equal to  $S$ . Each state of memory may be labeled with several or none  $r$ -occurring sets, but each  $r$ -occurring set must exist as a label to some state of memory (by definition). Let  $\mathcal{U}_x$  be the collection of  $r$ -occurring labels for memory state  $x$ . We want to upper bound the size of  $\mathcal{U}_x$ . Following lemma along with (Even, Odd)-town Lemma (Corollary 4) helps us in doing exactly that.

► **Lemma 13.** *If  $S_1$  and  $S_2$  belong to  $\mathcal{U}_x$ , then  $|S_1 \setminus S_2|$  is odd.*

**Proof.** Let  $D = S_1 \Delta S_2$ , let  $D_1 = S_1 \setminus S_2$ , and let  $D_2 = S_2 \setminus S_1$ . Assume to the contrary that  $|D_1|$  is even. Note then that  $|D_2|$  is also even (since  $|S_1| = |S_2| = 2r$ ) and that  $|D| = 2|D_1|$ . We'll consider two possible cases for the state of the game after turn  $2r$ : 1. Alice is at state  $x$ , and the set of numbers that have been said is  $S_1$ , and 2. Alice is at state  $x$ , and the set of numbers that have been said is  $S_2$ .

Consider the following strategy that Bob can play in either of these cases: “say the smallest number that has not been said and is not in  $D$ ”. We claim that if Bob uses this strategy, Alice will be the first person (after turn  $2r$ ) to say an element of  $D$ . Note that Bob will not say an element of  $D$  until after turn  $2n - |D|/2$ ; this is since:

1. If we are in case 1, then all of the numbers in  $D_1$  have been said but none of the numbers in  $D_2$  have been said. There are therefore  $|D_2| = |D|/2$  numbers in  $D$  which have not been said, so Bob can avoid saying an element of  $D$  till turn  $2n - |D|/2$ .
2. Likewise, if we are in case 2, the argument proceeds symmetrically.

On the other hand, if no element of  $D$  is spoken by either player between turn  $2r$  and turn  $2n - |D|/2$ , then at turn  $2n - |D|/2 + 1$ , the only remaining elements belong to  $D$  (the set of remaining elements is either  $D_1$  or  $D_2$ , depending on which case we are in). If  $|D_1| = |D|/2$  is even, then it is Alice's turn to speak at turn  $2n - |D|/2 + 1$ . It follows that Alice will be the first person after turn  $r$  to say an element of  $D$ .

Let  $y_1$  be the memory state of Alice when she first speaks an element of  $D$  in case 1, and define  $y_2$  similarly. We claim that  $y_1 = y_2$ . Indeed, since Alice's strategy is deterministic and starts from  $x$  in both cases 1 and 2, Bob's strategy plays identically in both case 1 and case 2 until an element of  $D$  has been spoken. It follows that Alice must speak the same element of  $D$  in both cases. But if this element is in  $D_1$ , and they are in case 1, then this element has already been said before; similarly, if this element is in  $D_2$ , and they are in case 2, then this element has also been said before. Regardless of which element in  $D$  Alice speaks at this state, there is some case where she loses, which contradicts the fact that Alice's strategy is successful. It follows that  $|D_1|$  must be odd, as desired. ◀

► **Claim 14.**  $|\mathcal{U}_x| \leq N$

**Proof.** We claim the sets in  $\mathcal{U}_x$  form an (Even, Odd)-town, from which this conclusion follows (Corollary 4). Each set in  $\mathcal{U}_x$  has cardinality  $2r$ , so all sets have even cardinality. By Lemma 13, any pair of distinct sets  $S_1, S_2 \in \mathcal{U}_x$  has odd  $|S_1 \setminus S_2|$ . Note that since  $|S_1 \cap S_2| = |S_1| - |S_1 \setminus S_2|$ , and since  $|S_1|$  is even, it follows that  $|S_1 \cap S_2|$  is odd, so any pair of sets have an odd cardinality intersection. ◀

Choose  $r = N/5$ . By Corollary 10,  $\mathcal{S}_r$  must have cardinality at least  $2^{(\log_2 5 - 2)N - o(N)}$ . Since each element in  $\mathcal{S}_r$  belongs to at least one collection  $\mathcal{U}_x$ , and since each collection  $\mathcal{U}_x$  has cardinality at most  $N$  (Claim 14), the number  $M$  of memory states  $x$  is at least  $2^{(\log_2 5 - 2)N - o(N)} / (N + 1) = 2^{(\log_2 5 - 2)N - o(N)}$ . It follows that  $m = \log M \geq (\log_2 5 - 2)N - o(N)$ , as desired. This proves Theorem 11. ◀

## 4 Randomized Strategies for Alice

In this section, we consider randomized strategies for Alice. A *randomized strategy computable in memory  $m$*  is defined similarly as in Section 2, with the exception that the transition function no longer needs to be deterministic and need only be computable in  $RSPACE(m)$ ;

i.e. it must be computable by a space  $m$  Turing machine with access to a  $\text{poly}(N, m)$ -sized read-once random tape. We say a randomized strategy wins with probability  $p$  if Alice wins with probability at least  $p$  against every possible strategy for Bob.

Unfortunately, we do not know any randomized strategies with sublinear memory which win with high probability. We therefore relax the definition of randomized strategy above and consider randomized strategies where Alice has oracle access to a perfect matching  $M$  chosen uniformly at random from all perfect matchings on  $K_N$  where  $K_N$  is the complete graph on  $N$  vertices (we assume here that  $N$  is even). Calling this oracle with  $x \in [N]$  returns  $x$ 's match  $M(x)$  in this matching, and  $M$  may be called arbitrarily many times during the computation of  $f$ .

One way to interpret the following upper bounds is as a source of difficulty for proving strong lower bounds for randomized strategies (of the form “you need linear space to succeed with high probability”), since a randomized strategy with memory  $m$  with access to a random matching can be viewed as a convex combination of deterministic strategies weakly computable in memory  $m$ . This means that any attempt to extend aforementioned strong lower bounds against randomized strategies must fail against this stronger class of randomized strategies.

Another way to interpret this model of computation is as a specific case of the setting where Alice has arbitrary read access to her random tape (as opposed to read-once access). It is known [11] that having arbitrary read access to randomness is more powerful than read-once access as long as certain probabilistic space classes do not collapse. Proving a strong lower bound for randomized strategies would provide more evidence for this separation (this time in the setting of streaming games).

We begin by showing that with only logarithmic space (and access to a random perfect matching), Alice can already win with probability  $\Omega(1/N)$ . In contrast, without access to a random matching, we know no logarithmic space strategy that succeeds with better than an exponentially small probability.

► **Theorem 15.** *When  $N$  is even, there exists a randomized strategy (with access to a uniformly random perfect matching  $M$  on  $K_N$ ) for Alice with space complexity  $O(\log N)$  which succeeds with probability  $\Omega(1/N)$ .*

**Proof.** Consider the following strategy for Alice. She begins by sampling a uniform element  $x$  from  $[N]$ . On her first turn, Alice says  $x$ . On subsequent turns, if Bob has just previously said  $y$ , Alice replies with  $M(y)$ .

Note that with this strategy, Alice wins playing against Bob if the last number said by Bob on turn  $N$  is  $M(x)$ . This follows since  $M$  is a matching, so there is no other number  $y \neq M(x)$  that Bob can say where  $M(y)$  has also already been said. It follows that if Bob says  $M(x)$  at turn  $N$ , then Alice is guaranteed to win.

What is the probability Bob says  $M(x)$  before turn  $N$ ? We will show it is less than  $1 - 1/N$ . To do this, we first claim that any winning strategy for Bob (i.e. any strategy that doesn't repeat previously said elements) has the same probability of saying  $M(x)$  before turn  $N$ . This follows from symmetry: since a uniformly random perfect matching conditioned on containing a submatching is still a uniformly random perfect matching on the remaining vertices, at any point in the protocol, if  $M(x)$  has not been said yet, it has an equally likely chance of being any of the unsaid elements. Therefore, without loss of generality, assume Bob is playing according to the strategy where each turn he says the smallest number that has not been said so far.

Now, note that if  $M(x) = N$ , Bob will not say  $M(x)$  before turn  $N$  (there will always be a smaller unsaid element). But this happens with probability  $1/N$ , and therefore Alice succeeds with probability at least  $1/N$ . ◀

We will next show how to extend this idea to construct an  $\tilde{O}(\sqrt{N})$  strategy for Alice which succeeds with high probability. To do this, we will need the following folklore result on determining missing elements from a set in a streaming setting.

► **Definition 16.** The “missing  $k$  numbers problem” is a streaming problem where a subset  $S$  of cardinality  $N - k$  is chosen from  $[N]$ , and Alice is shown the elements of  $S$  one at a time, in some order. Alice’s goal is to output the set  $[N] \setminus S$ .

► **Lemma 17.** *There exists an  $O(k \log N)$ -space deterministic algorithm for the missing  $k$  numbers problem.*

**Proof.** See [10]. For completeness, we include the proof here.

Choose a prime  $q \in (N, 2N]$ . We will perform all subsequent computations over the field  $\mathbb{F}_q$ . For a subset  $T$  of  $[N]$ , define  $p_i(T) = \sum_{x \in T} x^i$ . We first claim that if  $|T| \leq k$ , then Alice can recover the elements of  $T$  given the values of  $p_i(T)$  for  $1 \leq i \leq k$ .

To show this, define  $e_i(T)$  to be the value of  $i$ th elementary symmetric polynomial over the elements of  $T$  (that is,  $e_i(T) = \sum_{T' \subseteq T, |T'|=i} \prod_{x \in T'} x$ ). Newton’s identities allow us to compute (in space  $O(\log N)$ ) the values of  $e_i(T)$  for  $1 \leq i \leq k$  from the values of  $p_i(T)$  for  $1 \leq i \leq k$  (since  $k < N < q$ , all of these operations are valid over  $\mathbb{F}_q$ ). Now, each element  $x$  of  $T$  is a root of the polynomial

$$x^k - e_1(T)x^{k-1} + e_2(T)x^{k-2} - \dots + (-1)^k e_k(T) = 0.$$

We can factor this in space  $O(\log N)$  by trying all possible  $x \in [N]$ , and therefore recover the original set  $T$ .

To solve the missing  $k$  numbers problem, Alice first computes  $p_i(S)$  for each  $1 \leq i \leq k$  using  $O(k \log N)$  space, updating each count every time she receives a new element from  $S$ . This allows her to compute  $p_i([N] \setminus S)$  via  $p_i([N] \setminus S) = p_i([N]) - p_i(S)$ . Finally, since  $|[N] \setminus S| = k$ , she can use the previous algorithm to recover the set  $[N] \setminus S$ . ◀

► **Theorem 18.** *When  $N$  is even, there exists a randomized strategy (with access to a uniformly random perfect matching  $M$  on  $K_N$ ) for Alice with space complexity  $O(\sqrt{N} \log^2 N)$  which succeeds with probability  $1 - O(1/N)$ .*

**Proof.** Consider the following strategy for Alice. Alice begins by sampling  $r = \sqrt{N}$  distinct elements  $X = \{x_1, x_2, \dots, x_r\}$  uniformly at random from all  $\binom{N}{r}$  subsets of  $r$  elements of  $N$  (and stores them in memory).

At every point in the game, Alice uses her  $O(\sqrt{N} \log^2 N) = O(r \log^2 N)$  bits of memory to keep track of: i) which elements of  $X$  have already been said, and ii) the first  $k = r \log N$  power sums of the elements said so far, modulo some prime  $q \in (N, 2N]$  (as per the proof of Lemma 17).

Alice begins by choosing a random element of  $X$  and saying it. Her algorithm in later rounds is defined as follows:

1. If less than or equal to  $k$  elements remain unchosen (i.e., this is turn  $N - k$  or later), Alice uses the  $O(k \log n)$ -space algorithm from Lemma 17 to compute the remaining unsaid elements. She then says one of these elements, chosen arbitrarily.
2. Otherwise, if Bob has just said  $y$ , Alice checks if  $M(y)$  belongs to  $X$ .
  - If it does not belong to  $X$ , Alice says  $M(y)$ .



## 36:10 The Space Complexity of Mirror Games

- If  $M(y)$  belongs to  $X$  but has not been said yet, Alice says  $M(y)$  and marks down  $M(y)$  as having been said.
- If  $M(y)$  belongs to  $X$  and has already been said, Alice chooses an element of  $X$  uniformly at random from the elements of  $X$  which have not yet been said (if there are no such elements, Alice gives up and says a random element). She then marks this element as having been said.

Let  $T$  be the random variable denoting the first turn in which all of the elements of  $X$  have been said. We first claim that if  $T \geq N - k$ , then Alice succeeds. Indeed, since the algorithm of Lemma 17 always succeeds, Alice is guaranteed to succeed if she makes it to turn  $N - k$ . On the other hand, the only way for Alice to fail before turn  $N - k$  is if all of the elements of  $X$  have already been said, and Alice is forced to say a random element.

We now claim that with high probability,  $T \geq N - k$ . To see this, note that, similarly as in the proof of Theorem 15, any non-trivial strategy of Bob will give rise to the same distribution over  $T$ . This follows from the fact that at any point in the protocol, each unsaid element has the same probability of belonging to the set  $X$ .

We can therefore assume that Bob is playing under the strategy where on his turn, he says the smallest number that has not been said so far. Note that under this strategy, Bob will only say numbers less than or equal to  $i$  on turn  $i$ ; it follows that if there exists any  $x_i$  such that  $x_i \geq N - k$ , then also  $T \geq N - k$ .

We can thus compute

$$\begin{aligned}
 \Pr[T < N - k] &\leq \Pr[\forall i, x_i \leq N - k] \\
 &= \frac{\binom{N-k}{r}}{\binom{N}{r}} \\
 &\leq \left(\frac{N-k}{N}\right)^r \\
 &= \left(1 - \frac{\log N}{\sqrt{N}}\right)^{\sqrt{N}} \\
 &= O(1/N).
 \end{aligned}$$

It follows that Alice wins with probability at least  $1 - O(\frac{1}{N})$ , as desired. ◀

Note that the same upper bound holds when Alice does not have an oracle access to a random matching but Bob's strategy is to choose a random number in  $[N]$  in every turn.

### 5 The $(a, b)$ -Mirror Game

Consider the following generalization of the mirror game, where Alice says  $a$  numbers each turn, and Bob says  $b$  numbers each turn. We call this new game the  $(a, b)$ -mirror game.

As with the regular  $(1, 1)$ -mirror game, mirror strategies exist for the class of  $(1, b)$ -mirror games (and similarly, the  $(a, 1)$ -mirror games).

► **Theorem 19.** *If  $N$  is divisible by  $b + 1$ , then Bob has a winning strategy computable in  $O(\log N)$  memory for the  $(1, b)$ -mirror game.*

**Proof.** Divide the  $N$  elements of  $[N]$  into  $N/(b+1)$  consecutive  $(b+1)$ -tuples. Any time Alice says an element in a  $(b+1)$ -tuple, Bob says all the remaining elements in that  $(b+1)$ -tuple. ◀

Unlike in the  $(1,1)$ -mirror game, we cannot rule out the existence of low space winning strategies for other games and other choices of  $N$ . In this section, we summarize what we know about low-space winning strategies for  $(a,b)$ -mirror games.

### 5.1 $a + b$ is Prime

We begin by considering the set of  $(a,b)$ -mirror games where  $a + b$  is a prime,  $p$ . The Frankl-Wilson Lemma (Lemma 5) allows us to extend some of our proof techniques from Theorem 11 to this case.

In the case where  $a = 1$ , we have the following analogue to Theorem 11, fully characterizing the  $N$  where Bob has a low space winning strategy (Theorem 19) and where Bob requires linear space.

► **Theorem 20.** *Let  $p$  be a prime. If  $N$  is not divisible by  $p$ , then any winning strategy for Bob in the  $(1, p - 1)$ -mirror game requires  $\Omega(N)$  space.*

**Proof.** See Appendix A. ◀

When  $a > 1$ , our proof techniques no longer allow us completely characterize the set of  $N$  where Bob requires  $\Omega(N)$  space to win. Instead, we only have the following partial characterization.

► **Theorem 21.** *Let  $p$  be a prime. If  $a + b = p$ , and  $N \bmod p \in \{a, a + 1, \dots, p - 1\}$ , then any winning strategy for Bob in the  $(a,b)$ -mirror game requires  $\Omega(N)$  space.*

**Proof.** See Appendix A. ◀

### 5.2 $a + b$ is Composite

The failure of the Frankl-Wilson lemma (Lemma 5) to hold modulo composite numbers prevents us from directly applying our proof techniques in this case. Gromulz's construction (Theorem 6) shows that there exist  $(m, L)$ -Modtown families containing a superpolynomial in  $N$  number of sets when  $m$  is composite.

However, a sufficiently small superpolynomial upper bound on the size of such a family would still allow us to prove the analogue of Theorem 20. In fact, any upper bound of  $F$  on the size of such a family leads to a lower bound of  $\log(2^{\Omega(n)}/F) = \Omega(n) - \log F$  on Bob's memory. This implies the following "contrapositive" to Theorem 20:

► **Theorem 22.** *If Bob has a  $o(N)$ -space winning strategy in a  $(1, m - 1)$  game when  $N \equiv k \not\equiv 0 \pmod{m}$ , then there exists  $2^{\Omega(N)}$  sized matching vector families over  $\mathbb{Z}_m^N$  (for sufficiently large  $N$ ).*

**Proof.** See Appendix A. ◀

## 6 Open Problems

Our understanding of the space complexity aspects of mirror games is still very rudimentary. We conclude by listing some open problems we find interesting.

1. **When do low-space winning strategies exist?** Does either Alice or Bob ever have a low-space winning strategy for any  $(a,b)$  game when  $a$  and  $b$  are both larger than 1 (e.g., the  $(2,2)$  game)? Are there any cases where the best deterministic winning strategy

has space complexity strictly between  $O(\log N)$  and  $O(N)$  (e.g.  $O(\sqrt{N})$ ), or does the best winning strategy always fit into one of these two extremes? Is it ever the case that both Alice and Bob simultaneously have  $O(\log N)$  winning strategies?

2. **Beating low-space with low-space.** In order to show that Alice needs  $\Omega(N)$  space, our adversarial strategy for Bob also requires  $\Omega(N)$  space. Given a deterministic low-memory  $m$  strategy for Bob, is there a low-memory strategy for Alice which wins against it?
3. **Lower bounds for randomized strategies.** Can we prove any sort of lower bound against randomized strategies that win with high probability? What about against randomized strategies with additional power, such as access to a uniformly chosen matching or with multiple read access to the random tape? Is our upper bound of  $\tilde{O}(\sqrt{N})$  tight in the latter contexts?
4. **Composite  $a + b$ .** Is it possible to show some sort of converse to Theorem 22, that any large enough matching vector family gives rise to a low space strategy for Bob? Or is it possible to show linear space lower bounds for composite  $a + b$  via some other approach?
5. **More general mirror games.** One of the great successes of the theory of combinatorial games [2] is that its techniques apply to essentially all sequential two-player games with perfect information. Is there a more general class of games beyond the family of  $(a, b)$ -mirror games with similar space complexity phenomena? One possible other family of such games with mirror strategies comes from a generalization of the combinatorial game “Cram”, where Alice and Bob take turn placing dominoes (more generally, an element of some set of “symmetric” polyominoes) onto a  $w$  by  $h$  grid (more generally, some high dimensional grid, or any “symmetric” subset of a high-dimensional grid). For example, the original mirror game can be thought of as Cram on a 1 by  $N$  board, where players take turn placing unit squares. Can we say anything about the space complexity of playing Cram, or its generalizations?

---

## References

- 1 Sanjeev Arora and Boaz Barak. *Computational complexity: a modern approach*. Cambridge University Press, 2009.
- 2 Elwyn R Berlekamp, John Horton Conway, and Richard K Guy. *Winning ways for your mathematical plays*, volume 3. AK Peters Natick, 2003.
- 3 ER Berlekamp. On subsets with intersections of even cardinality. *Canad. Math. Bull.*, 12(4):471–477, 1969.
- 4 Abhishek Bhowmick, Zeev Dvir, and Shachar Lovett. New bounds for matching vector families. *SIAM Journal on Computing*, 43(5):1654–1683, 2014.
- 5 Zeev Dvir, Parikshit Gopalan, and Sergey Yekhanin. Matching vector codes. *SIAM Journal on Computing*, 40(4):1154–1178, 2011.
- 6 Zeev Dvir and Sivakanth Gopi. 2-Server PIR with Subpolynomial Communication. *Journal of the ACM (JACM)*, 63(4):39, 2016.
- 7 Klim Efremenko. 3-query locally decodable codes of subexponential length. *SIAM Journal on Computing*, 41(6):1694–1703, 2012.
- 8 Peter Frankl and Richard M. Wilson. Intersection theorems with geometric consequences. *Combinatorica*, 1(4):357–368, 1981.
- 9 Vince Grolmusz. Superpolynomial size set-systems with restricted intersections mod 6 and explicit Ramsey graphs. *Combinatorica*, 20(1):71–86, 2000.
- 10 S Muthukrishnan. Data Streams: Algorithms and Applications (Foundations and Trends in Theoretical Computer Science). *Foundations and Trends in Theoretical Computer Science*, 2005.

- 11 Noam Nisan. On read-once vs. multiple access to randomness in logspace. In *Structure in Complexity Theory Conference, 1990, Proceedings., Fifth Annual*, pages 179–184. IEEE, 1990.
- 12 Tibor Szabó. <http://discretemath.imp.fu-berlin.de/DMII-2011-12/linalgmethod.pdf>, 2011.

## A

 Omitted Proofs

### Proof of Theorem 20

Theorem 20 is a special case of Theorem 21 setting  $a = 1$ . Hence, we go on to prove that.

### Proof of Theorem 21

**Proof.** We proceed similarly to the proof of Theorem 11. As before, call a subset  $S$  of  $[N]$   $r$ -occurring if it is possible that immediately after turn  $2r$ , the set of numbers that have been said is equal to  $S$ . Let  $\mathcal{S}_r$  be the collection of all  $r$ -occurring sets. As before,  $\mathcal{S}_r$  is  $(p, r)$ -covering and we can choose  $r$  such that  $|\mathcal{S}_r| \geq 2^{\Omega(N)}$ .

Write  $N = pm + k$ ,  $k \in \{a, a + 1, \dots, p - 1\}$ , and fix a value  $r \in [n]$ . For a memory state  $x$  out of the  $M$  possible memory states of Bob and an  $r$ -occurring set  $S$ , label  $x$  with  $S$  if it is possible that Bob is at memory state  $x$  when the set of numbers that have been said is equal to  $S$ . Each state of memory may be labelled with several  $r$ -occurring sets or none, but each  $r$ -occurring set must exist as a label to some state of memory (by definition). Let  $\mathcal{U}_x$  be the collection of  $r$ -occurring labels for memory state  $x$ .

► **Lemma 23.** *If  $S_1$  and  $S_2$  belong to  $\mathcal{U}_x$ , then  $|S_1 \setminus S_2| \bmod p \in \{k - a + 1, \dots, k - 1, k, \dots, p - 1\}$ .*

**Proof.** Let  $D = S_1 \Delta S_2$ , let  $D_1 = S_1 \setminus S_2$ , and let  $D_2 = S_2 \setminus S_1$ . Assume to the contrary that  $|D_1| \equiv k_1 \pmod p$  where  $k_1 \in \{0, 1, \dots, k - a\}$ . Note then that  $|D_2| = |D_1|$  (since  $|S_1| = |S_2|$ ) and that  $|D| = 2|D_1|$ . We'll consider two possible cases for the state of the game after turn  $2r$  (each turn, either Alice says  $a$  numbers or Bob says  $b$  numbers): 1. Bob is at state  $x$ , and the set of numbers that have been said is  $S_1$ , and 2. Bob is at state  $x$ , and the set of numbers that have been said is  $S_2$ .

Consider the following strategy that Alice can play in either of these cases: “say the smallest number that has not been said that is not in  $D$ ”. We claim that if Alice uses this strategy, Bob will be the first person (after turn  $2r$ ) to say an element of  $D$ . Note that Alice will not say an element of  $D$  until turn  $2\lfloor \frac{N - |D_1|}{p} \rfloor + 1$ ; this is since:

1. If we are in case 1, then all of the numbers in  $D_1$  have been said but none of the numbers in  $D_2$  have been said. There are therefore  $|D_2| = |D_1|$  numbers in  $D$  which have not been said, so Alice can avoid saying an element of  $D$  until turn  $2\lfloor \frac{N - |D_1|}{p} \rfloor + 1$  as the number of numbers remaining before Alice's turn would be  $|D_1| + k - k_1 \geq |D_1| + a$ . Hence, Alice can say  $a$  numbers not belonging to the set  $D_1$ .
2. Likewise, if we are in case 2, the argument proceeds symmetrically.

On the other hand, if no element of  $D$  is spoken by either player between turn  $2r$  and turn  $2\lfloor \frac{N - |D_1|}{p} \rfloor + 1$ , then at turn  $2\lfloor \frac{N - |D_1|}{p} \rfloor + 2$ , Bob has to say  $b$  elements from  $|D_1| + k - k_1 - a$  elements and hence, there would be intersection with elements of  $D$  (as  $k - k_1 - a \leq p - 1 - a = b - 1$ ).

Let  $y_1$  be the memory state of Bob when he first speaks an element of  $D$  in case 1, and define  $y_2$  similarly. We claim that  $y_1 = y_2$ . Indeed, since Bob's strategy is deterministic and starts from  $x$  in both cases 1 and 2, Alice's strategy plays identically in both case 1 and case

2 until an element of  $D$  has been spoken. It follows that Bob must speak the same element of  $D$  in both cases. But if this element is in  $D_1$ , and they are in case 1, then this element has already been said before; similarly, if this element is in  $D_2$ , and they are in case 2, then this element has also been said before. Regardless of which element in  $D$ , Bob speaks at this state, there is some case where he loses, which contradicts the fact that Bob's strategy is successful. It follows that  $|D_1| \bmod p \in \{k - a + 1, \dots, k - 1, k, \dots, p - 1\}$ , as desired. ◀

► **Claim 24.**  $|\mathcal{U}_x| \leq p \binom{N}{p-1}$ .

**Proof.** We claim the sets in  $\mathcal{U}_x$  form a  $(p, \{1, 2, \dots, p-1\})$ -Modtown, from which this conclusion follows (Lemma 5). Each set in  $\mathcal{U}_x$  has cardinality  $pr \equiv 0 \pmod p$ . By Lemma 23, any pair of distinct sets  $S_1, S_2 \in \mathcal{U}_x$  has  $|S_1 \setminus S_2| \bmod p \in \{k - a + 1, \dots, k - 1, k, \dots, p - 1\} \subseteq \{1, 2, \dots, p - 1\}$  (as  $k \geq a$ ). Note that since  $|S_1 \cap S_2| = |S_1| - |S_1 \setminus S_2|$ , it follows that  $|S_1 \cap S_2| \bmod p$  also belongs to  $\{1, 2, \dots, p - 1\}$ . ◀

We established that for some  $r$ ,  $\mathcal{S}_r$  must have cardinality at least  $2^{\Omega(N)}$ . Since each element in  $\mathcal{S}_r$  belongs to at least one collection  $\mathcal{U}_x$ , and since each collection  $\mathcal{U}_x$  has cardinality at most  $pN^p$  (Claim 24), the number  $M$  of memory states  $x$  is at least  $2^{\Omega(N)}/(pN^p)$ . It follows that  $m = \log M \geq \Omega(N)$ , as desired. This proves Theorem 21. ◀

## Proof of Theorem 22

**Proof.** In proof of Theorem 21, we use the fact that  $p$  is prime only to bound the size of  $\mathcal{U}_x$  and thus, Lemma 23 ( $a = 1$ ) hold even for composite  $m$ . As  $\mathcal{S}_r$  must have cardinality at least  $2^{\Omega(N)}$  and there are at most  $2^{o(N)}$  memory states  $x$ , by pigeonhole principle, there exists  $x'$  with  $|\mathcal{U}_{x'}| \geq 2^{\Omega(N)}$ . Let's define the set of vectors  $U$  based on  $\mathcal{U}_{x'}$ . For every subset  $S \in \mathcal{U}_{x'}$ , we add a vector  $v_S \in \{0, 1\}^N$  to  $U$  where  $v_S$  is the characteristic vector of  $S$  ( $v_S(l) = 1 \iff l \in S$ ). Clearly,  $|U| = |\mathcal{U}_{x'}|$  and  $U \subseteq \{0, 1\}^N$ . We claim the  $U$  and  $U$  form a Matching Vector family as  $v_S^T v_S = |S| \bmod m = 0$ , for all  $S$  and  $v_{S_1}^T v_{S_2} = |S_1 \cap S_2| \not\equiv 0 \pmod m$ ,  $\forall S_1 \neq S_2$  by Lemma 23 (when  $N \equiv k \not\equiv 0 \pmod m$ ). Hence, we have a exponential sized Matching Vector family  $(U, U)$  over  $\mathbb{Z}_m^N$ . ◀

# The Subgraph Testing Model

Oded Goldreich

Department of Computer Science, Weizmann Institute of Science, Rehovot, Israel  
oded.goldreich@weizmann.ac.il

Dana Ron

School of Electrical Engineering, Tel Aviv University, Tel Aviv, Israel  
danaron@tau.ac.il

---

## Abstract

---

We initiate a study of testing properties of graphs that are presented as subgraphs of a fixed (or an explicitly given) graph. The tester is given free access to a base graph  $G = ([n], E)$ , and oracle access to a function  $f : E \rightarrow \{0, 1\}$  that represents a subgraph of  $G$ . The tester is required to distinguish between subgraphs that possess a predetermined property and subgraphs that are far from possessing this property.

We focus on bounded-degree base graphs and on the relation between testing graph properties in the subgraph model and testing the same properties in the bounded-degree graph model. We identify cases in which testing is significantly easier in one model than in the other as well as cases in which testing has approximately the same complexity in both models. Our proofs are based on the design and analysis of efficient testers and on the establishment of query-complexity lower bounds.

**2012 ACM Subject Classification** Theory of computation  $\rightarrow$  Streaming, sublinear and near linear time algorithms

**Keywords and phrases** Property Testing, Graph Properties

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2019.37

**Funding** This research was partially supported by the Israel Science Foundation (grants No. 671/13 and 1146/18).

## 1 Introduction

Property testing refers to probabilistic algorithms with sub-linear complexity for deciding whether a given object has a predetermined property or is far from any object having this property. Such algorithms, called testers, obtain local views of the object by *performing queries* and their performance guarantees are stated with respect to a distance measure that (combined with a distance parameter) determines what objects are considered far from the property.

In the last couple of decades, the area of property testing has attracted significant attention (see, e.g., [13, 31, 32, 14]). Much of this attention was devoted to testing graph properties in a variety of models ranging from the dense graph model [15], to the bounded-degree graph model [17], and to the sparse and general graph models [30, 24].<sup>1</sup> These models differ in two main parameters: the *types of queries* that potential testers can make, and the *distance measure* against which their performance is measured.

---

<sup>1</sup> These models are surveyed in Chapters 8, 9, and 10 of the textbook [14].



In all aforementioned models, the input graph is arbitrary, except for its size (and possibly its degree, in the case of the bounded-degree graph model). The same holds with respect to the graphs that are used to determine the distance of the input from the property. While some prior works (see, e.g., [3, 22, 6, 20, 9, 7, 29, 5]) restrict the input graph in certain natural ways, the restrictions considered so far were expressed in terms of general (“uniform”) graph properties (such as degree bound, hyperfiniteness, planarity, etc). See further discussion in Section 1.5.1.

In contrast, we envision circumstances in which the input is restricted to be a subgraph of some fixed graph that is known beforehand. For example, the fixed graph may represent an existing (or planned) network, and the subgraph represents the links that are actually in operation (or actually constructed). Alternatively, the graph may represent connections between data items that may exist under some known constraints, and the edges of the subgraph represent connections that actually exist. Either way, the input is a subgraph of some fixed graph, and the distance to having the property is measured with respect to subgraphs of the same fixed graph.

## 1.1 The model

In accordance with the foregoing discussion, in the **subgraph testing model**, there is a fixed **base graph**, denoted  $G = ([n], E)$ , and the tester is given oracle access to a function  $f : E \rightarrow \{0, 1\}$  that represents a subgraph of  $G$  in the natural manner (i.e.,  $f$  represents the subgraph  $([n], \{e \in E : f(e) = 1\})$ ). Alternatively, the base graph  $G$  is not fixed, but the tester is given free access to  $G$ .

► **Definition 1.1** (subgraph tester). Fixing  $G = ([n], E)$  and  $\Pi_G \subseteq \mathcal{F}_G \stackrel{\text{def}}{=} \{f : E \rightarrow \{0, 1\}\}$ , a **subgraph tester** for  $\Pi_G$  is a probabilistic oracle machine, denoted  $T$ , that, on input a (proximity) parameter  $\epsilon$ , and oracle access to a function  $f : E \rightarrow \{0, 1\}$ , outputs a binary verdict that satisfies the following two conditions.

1.  $T$  *accepts inputs in*  $\Pi_G$ : For every  $\epsilon > 0$ , and for every  $f \in \Pi_G$ , it holds that  $\Pr[T^f(\epsilon) = 1] \geq 2/3$ .
2.  $T$  *rejects inputs that are  $\epsilon$ -far from*  $\Pi$ : For every  $\epsilon > 0$ , and for every  $f : E \rightarrow \{0, 1\}$  that is  $\epsilon$ -far from  $\Pi_G$  it holds that  $\Pr[T^f(\epsilon) = 0] \geq 2/3$ , where  $f$  is  $\epsilon$ -far from  $\Pi_G$  if for every  $h \in \Pi_G$  it holds that  $|\{e \in E : f(e) \neq h(e)\}| > \epsilon \cdot |E|$ .

If the first condition holds with probability 1 (i.e.,  $\Pr[T^f(\epsilon) = 1] = 1$  for  $f \in \Pi_G$ ), then we say that  $T$  has **one-sided error**; otherwise, we say that  $T$  has **two-sided error**.

In the alternative formulation, the subgraph tester is given  $G$  as an explicit input (along with  $\epsilon$ ). In this case, the random variable being considered is  $T^f(G, \epsilon)$ .

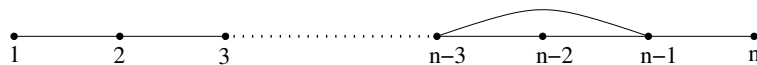
Definition 1.1 falls within the framework of massively parameterized property testing (cf. [28]). The massive parameter is the base graph  $G = ([n], E)$ , and the actual input is a function  $f : E \rightarrow \{0, 1\}$  (which represents a subgraph of  $G$ ).

(The subgraph testing model is syntactically identical to the *orientation model* [21], but semantically these models are fundamentally different; see further discussion in Section 1.5.2.)

As usual in the area, our primary focus is on the query complexity of such testers, and our secondary focus is on their time complexity. Both complexities are stated as a function of the proximity parameter  $\epsilon$  and the base graph  $G$ . Indeed, the dependency of these complexities on  $G$ , or rather on some parameters of  $G$ , will be of major interest.

As an illustration, consider the problem of testing whether the subgraph is bipartite. If the base graph is bipartite, then this problem is trivial (since every subgraph is bipartite).





■ **Figure 1** For  $n \geq 6$ , the  $n$ -vertex path is oriented by the additional edge  $\{n-3, n-1\}$ .

If the base graph is  $\mathcal{M}$ -minor free<sup>2</sup>, for any fixed family of graphs  $\mathcal{M}$ , then testing (with distance parameter  $\epsilon$ ) can be done in  $\text{poly}(1/\epsilon)$ -time (see Proposition 2.2). Lastly, if the base graph is of bounded-degree, then testing can be done in  $\text{poly}(1/\epsilon) \cdot \tilde{O}(\sqrt{n})$ -time (see Theorem 1.2), and this result is optimal in general (i.e., for arbitrary bounded-degree base graphs, see Part 1 of Theorem 1.4).

Our main focus will be on the case that the base graph is sparse (e.g., of bounded-degree). Furthermore, we shall be interested in cases in which the subgraph testing model is different from other testing models. Still, let us make a couple of comments regarding cases in which the subgraph testing model coincides with other testing models.

**The dense graph model is a special case of subgraph testing.** For the base graph  $G = K_n$  (i.e., the  $n$ -vertex clique), the subgraph testing model coincides with the dense graph model. This is the case since adjacency queries (as in the dense graph model) correspond to edges of the base graph  $G$ , and the distance measure used in both models is the same.

**General property testing as a special case of subgraph testing.** If the base graph  $G$  is sparse and asymmetric (i.e., its automorphism group consists solely of the identity permutation), then the subgraph testing model captures property testing (for Boolean functions) at large. This is shown as follows.

For  $n \geq 6$ , consider an  $n$ -vertex graph  $G'$  consisting of an  $n$ -vertex long path augmented with the edge  $\{n-3, n-1\}$  (see Figure 1). Observe that the only automorphism of this graph is the identity permutation, and augment  $G'$  with self-loops on each of the  $n$  vertices, deriving a base graph  $G$  with  $2n$  edges. (We note that the construction can be modified so that self-loops are avoided, by replacing them with disjoint cycles of length 3.) Lastly, associate any function  $f : [n] \rightarrow \{0, 1\}$  with a subgraph of  $G$  that contains  $G'$  as well as the self-loop on vertices in  $f^{-1}(1)$ . Note that, by the asymmetry of  $G'$ , there is a bijection between the set of Boolean functions over  $[n]$  and the subgraphs of  $G$  that contain  $G'$ , and that distances between the two models are preserved up to a factor of 2.<sup>3</sup>

**On our terminology: Testing graph properties in the subgraph model.** Unless the base graph  $G = ([n], E)$  is closed under all possible relabelings of  $[n]$  (which happens if and only if  $G$  is either the complete graph or the empty graph),<sup>4</sup> we cannot expect a (non-empty) set of

<sup>2</sup> Recall that a graph  $M$  is a minor of graph  $G$  if  $M$  can be obtained from  $G$  by vertex deletions, edge deletions and edge contractions; a graph  $G$  is  $\mathcal{M}$ -minor free for a family of graphs  $\mathcal{M}$ , if no graph in  $\mathcal{M}$  is a minor of  $G$ .

<sup>3</sup> The argument extends to any sparse graph  $G'$  that is asymmetric. Recall that almost all (bounded degree) graphs are asymmetric (cf. [10, 25]). On the other hand, an asymmetric graph cannot contain two isolated vertices, and thus it must contain at least a linear number of edges.

<sup>4</sup> Indeed, the  $n$ -vertex complete graph and the empty ( $n$ -vertex) graph are closed under all possible relabelings of  $[n]$ . On the other hand, if  $G$  is neither the complete graph nor the empty graph, then  $G$  is not closed under all possible relabelings of  $[n]$ . To see this observe that  $G$  must contain a vertex  $w$  that has degree in  $[n-2]$ ; that is, its neighbor set, denoted  $\Gamma_G(w)$ , is neither empty nor contains all other vertices in  $G$ . Picking  $u \in \Gamma_G(w)$  and  $v \notin \Gamma_G(w)$ , observe that the permutation  $\pi$  that switches  $u$

its subgraphs,  $\Pi_G$ , to constitute a graph property (i.e., to be closed under all relabelings of  $[n]$ ). That is, in the general case, the property  $\Pi_G \subseteq \mathcal{F}_G$  is not a graph property, since it is not closed under isomorphism (because  $\mathcal{F}_G$  is not closed under isomorphism). Nevertheless, for any base graph  $G$  and every graph property  $\Pi$ , we shall refer to  $\Pi_G = \mathcal{F}_G \cap \Pi$  as a graph property.

Throughout this paper, we assume that  $G$  contains no isolated vertices. This can be assumed without loss of generality, because, for every graph  $G'$  that is obtained from the graph  $G = ([n], E)$  by adding isolated vertices, it holds that  $\mathcal{F}_G = \mathcal{F}_{G'}$ , since in both cases the subgraphs are represented by Boolean functions on the same edge-set (i.e.,  $E$ ).

## 1.2 Results

Throughout this paper, the base graph  $G$  is viewed as a varying parameter, which may grow when other parameters (e.g., the degree bound  $d$ ) are fixed. We focus on bounded-degree base graphs and on the relation between testing graph properties in the subgraph model and testing the same properties in the bounded-degree graph (BDG) model.

Recall that in the BDG model [17], the tester is explicitly given three parameters:  $n$ ,  $d$ , and  $\epsilon$ . Its goal is to distinguish with probability at least  $2/3$  between the case that a graph  $G = ([n], E)$  (of maximum degree bounded by  $d$ ) has a prespecified property  $\Pi$ , and the case that  $G$  is  $\epsilon$ -far from having the property  $\Pi$ , where a graph is said to be  $\epsilon$ -far from having  $\Pi$  if more than  $\epsilon \cdot d \cdot n$  edge modifications (additions or removals) are required in order to obtain a graph (of maximum degree bounded by  $d$ ) that has  $\Pi$ . To this end the tester can perform queries of the form “who is the  $i^{\text{th}}$  neighbor of vertex  $v$ ?”, for  $v \in [n]$  and  $i \in [d]$ .<sup>5</sup> Unless stated explicitly otherwise, the degree bound  $d$  is a constant.

Obviously, the relationship between the subgraph model and the BDG model depends on the property being tested as well as on the base graph used in the subgraph model. We identify cases in which testing is significantly easier in one model than in the other as well as cases in which testing has approximately the same complexity in both models.

More specifically, we distinguish *downward-monotone* graph properties from other graph properties, where a graph property is called *downward-monotone* if it is preserved under omission of edges (i.e., if  $G = ([n], E)$  has the property, then so does  $G' = ([n], E')$  for every  $E' \subset E$ ).

For each of the theorems stated in this section, we provide either a proof outline or a proof idea in Section 1.3. Full proofs of Theorems 1.2, 1.3, and 1.7 (as well as Propositions 1.6 and 2.1) appear in Section 2. The proofs of our other results can be found in our technical report [19].

### 1.2.1 Downward-monotone properties

We first observe that, for every bounded-degree graph  $G = ([n], E)$  and any downward-monotone graph property  $\Pi$ , testing  $\Pi \cap \mathcal{F}_G$  in the subgraph model (w.r.t. the base graph  $G$ ) reduces to testing  $\Pi$  in the BDG model.

► **Theorem 1.2** (a general upper bound on the complexity of testing downward-monotone properties (see Section 2.1)). *Let  $\Pi$  be a downward-monotone graph property that is testable with query complexity  $Q_d(\cdot, \cdot)$  in the bounded-degree graph model, where  $d \geq 2$  denotes the*

---

and  $v$ , while keeping all other vertices intact, does not preserve the graph  $G$  (i.e.,  $\pi(G) \neq G$ ).

<sup>5</sup> If  $v$  has less than  $i$  neighbors, then a special symbol is returned. It is sometimes assumed that the algorithm can also query the degree of any vertex of its choice, but this assumption saves at most a multiplicative factor of  $\log d$  in the complexity of the algorithm.

degree bound, and  $Q_d$  is a function of the proximity parameter and (possibly) the size of the graph. Then, for every base graph  $G = ([n], E)$  of degree  $d$ , testing whether a subgraph of  $G$  satisfies  $\Pi$  (with proximity parameter  $\epsilon$ ) can be done with query complexity  $d \cdot Q_d(\epsilon', n)$ , where  $\epsilon' = \epsilon/d$ . The same holds with respect to the time complexity. Furthermore, one-sided error is preserved.

(Note that, for constant  $d$ , it holds that  $\epsilon' = \Omega(\epsilon)$ .) Properties covered by Theorem 1.2 include bipartiteness, cycle-freeness, and all subgraph-freeness and minor-freeness properties. Hence, testers known for these properties in the BDG model (see [14, Chap. 9]) get translated to testers of similar complexity for the subgraph testing model.

While Theorem 1.2 asserts that testing downward-monotone graph properties in the subgraph model is not harder than testing these properties in the BDG model, it raises the question of whether the former task may be easier.

**Testing in the subgraph model may be trivial.** A trivial positive answer holds in case the base graph itself has the property (i.e.,  $G \in \Pi$ ). In this case, by the downward-monotonicity of  $\Pi$ , every subgraph of  $G$  has the property  $\Pi$ , which means that testing  $\Pi \cap \mathcal{F}_G$  in the subgraph model (w.r.t. the base graph  $G$ ) is trivial.

**Testing in the subgraph model may be easier (than in the BDG model).** A more interesting case in which testing in the subgraph model may be easier than in the BDG model occurs when the base graph is not in  $\Pi$ , but has some suitable property  $\Pi'$  that is not related to  $\Pi$ . In particular, if the base graph is  $\mathcal{M}$ -minor free, for some fixed set of graphs  $\mathcal{M}$ , then, for any downward-monotone graph property  $\Pi$ , testing  $\Pi \cap \mathcal{F}_G$  in the subgraph model has complexity that is independent of the size of the tested graph, whereas testing  $\Pi$  in the BDG model may require query complexity that depends on the size of the tested graph. More generally, we consider *hyperfinite* families of graphs [8], where an  $n$ -vertex graph  $G$  is  $t$ -hyperfinite for  $t : (0, 1) \rightarrow \mathbb{N}$  if, for every  $\epsilon > 0$ , removing at most  $\epsilon n$  edges from  $G$  results in a graph with no connected component of size exceeding  $t(\epsilon)$ . We mention that minor-free (bounded-degree) graphs are hyperfinite (with  $t(\epsilon) = O(1/\epsilon^2)$ ).

► **Theorem 1.3** (on the complexity of testing downward-monotone properties of subgraphs of hyperfinite base graphs (see Section 2.2)). *Let  $\Pi$  be a downward-monotone graph property and  $\mathcal{G}$  be a family of  $t$ -hyperfinite graphs. Then, for every bounded-degree base graph  $G = ([n], E)$  in  $\mathcal{G}$ , testing whether a subgraph of  $G$  satisfies  $\Pi$  can be done in time that depends only on the proximity parameter  $\epsilon$ . Furthermore, if  $\Pi$  is additive (i.e., a graph is in  $\Pi$  iff all its connected components are in  $\Pi$ ), then its query complexity is  $O(t(\epsilon/4)/\epsilon)$  and the tester has one-sided error.*<sup>6</sup>

Note that by Theorem 1.3, testing bipartiteness of subgraphs of any (bounded-degree) planar graph  $G$  has complexity  $\text{poly}(1/\epsilon)$ , whereas (by [17]) testing bipartiteness of  $n$ -vertex graphs in the BDG model has complexity  $\Omega(\sqrt{n})$ .<sup>7</sup>

**Testing in the subgraph model may not be easier (than in the BDG model).** On the other hand, there are cases in which the testers provided by Theorem 1.2 are essentially the best possible. Indeed, these cases correspond to base graphs that are not hyperfinite.

<sup>6</sup> In general, the tester has two-sided error and the query complexity is at most exponential in  $O(t(\epsilon/4)^2)$ .

<sup>7</sup> As discussed in Section 1.5.1, weaker results may be obtained by using testers for the BDG model that work under the corresponding promise.

► **Theorem 1.4** (testing  $c$ -colorability of subgraphs of general bounded-degree base graphs (see [19, Sec. 3])).

1. *There exist explicit bounded-degree graphs  $G = ([n], E)$  such that testing whether a subgraph of  $G$  is bipartite, with proximity parameter  $1/\text{poly}(\log |E|)$ , requires  $\tilde{\Omega}(\sqrt{|E|})$  queries.*
2. *There exist bounded-degree graphs  $G = ([n], E)$  such that testing whether a subgraph of  $G$  is 3-colorable, with constant proximity parameter, requires  $\Omega(|E|)$  queries.*

Item 2 asserts that the complexity of testing 3-Colorability in the subgraph model may be linear, just as in the BDG model. Item 1 should be contrasted with the tester obtained by applying Theorem 1.2 to the known tester for the BDG model [16]. The derived tester has complexity  $\text{poly}(1/\epsilon) \cdot \tilde{O}(\sqrt{|E|})$ , where  $\epsilon$  denotes the proximity parameter, whereas Item 1 implies that one cannot obtain better complexity in terms of  $\epsilon$  and  $|E|$  (e.g., complexity  $\text{poly}(1/\epsilon) \cdot Q(|E|)$  is possible only for  $Q(n) = \tilde{\Omega}(\sqrt{n})$ ).

### 1.2.2 Other properties (i.e., non downward-monotone properties)

When turning to graph properties that are not downward-monotone, the statement of Theorem 1.2 no longer holds. There exist graph properties that are significantly harder to test in the subgraph model than in the BDG model. Specifically:

► **Theorem 1.5** (testing in the subgraph model may be harder than in the BDG model (see [19, Sec. 4])). *There exists a property of graphs  $\Pi$  for which the following holds. On one hand,  $\Pi$  is testable in  $\text{poly}(1/\epsilon)$ -time in the bounded-degree graph model. On the other hand, there exist explicit graphs  $G = ([n], E)$  of constant degree such that testing whether a subgraph of  $G$  satisfies  $\Pi$  requires  $\Omega(\log \log n)$  queries. Furthermore, the property  $\Pi$  is (upwards) monotone, and the family of base graphs is hyperfinite.<sup>8</sup>*

The first part of the furthermore-clause implies that a result analogous to Theorem 1.2 does not hold for monotone (rather than downward-monotone) graph properties. The second part of the furthermore-clause implies that also a result analogous to Theorem 1.3 does not hold for monotone graph properties.

We comment that the property  $\Pi$  used in Theorem 1.5 is related to being Eulerian, and the base graphs are related to a cyclic grid. Hence, it is interesting to note that testing whether subgraphs of a cyclic grid are Eulerian can be done in complexity that only depends on the proximity parameters (see [19, Prop. 4.2]).

Turning back to monotone graph properties, we first note the trivial case in which the base graph  $G$  does not have the property (which implies that all its subgraphs lack this property as well). A non-trivial case is that of testing minimum degree (see Proposition 2.1). A more interesting case is that of connectivity.

► **Proposition 1.6** (testing connectivity in the subgraph model – see Section 2.1). *For every base graph  $G = ([n], E)$ , testing whether a subgraph of  $G$  is connected can be done in  $\text{poly}(1/\epsilon)$ -time.*

We mention that even the case of 2-edge connectivity, which has an efficient tester in the BDG model, seems challenging in the subgraph testing model (see Problem 1.9).

---

<sup>8</sup> See the definition of hyperfinite graphs preceding Theorem 1.3.

**A relatively general positive result.** We next state a result for a class of properties that are not downward-monotone (and not necessarily monotone either). This result is of the flavor of Theorem 1.2, but introduces an overhead that is logarithmic in the number of vertices. Specifically, we refer to the class of all graph properties that have proximity-oblivious testers of constant query complexity (in the BDG model) [18, Sec. 5]. We mention that such properties are “local” in the sense that satisfying them can be expressed as the conjunction of conditions that refer to constant-distance neighborhood in the graph (see Definition 2.4).

► **Theorem 1.7** (testing local properties in the subgraph model (see Section 2.3)). *Let  $\Pi$  be a local property and suppose that the base graph  $G$  is outerplanar and of bounded degree. Then, testing whether a subgraph of  $G = ([n], E)$  has property  $\Pi$  can be done using  $O(\epsilon^{-1} \log n)$  queries.*

The result stated in Theorem 1.7 extends to every graph having constant-size separating sets (the dependence on the size of the separating sets is given explicitly in Theorem 2.5).

**Testing in the subgraph model may be easier than in the BDG model.** Lastly we note that moving from the BDG model to the subgraph testing model makes the testing task *potentially* easier, since the subgraph tester knows *a priori* the possible locations of edges. This is reflected by the following result, which refers to any (bounded-degree) base graph.

► **Theorem 1.8** (a property that is extremely easier in the subgraph model). *For every constant  $d$ , there exists a graph property  $\Pi_d$  that requires linear query complexity in the bounded-degree model but can be tested using  $O(1/\epsilon)$  queries in the subgraph model w.r.t. every base graph of maximum degree  $d$ .*

Since the proof of Theorem 1.8 is short and simple, we provide it next.

**Proof.** Fixing  $d$ , let  $\Pi_d$  be a set of  $d$ -regular graphs such that testing  $\Pi_d$  in the BDG model (with degree bound  $d$ ) requires a linear number of queries (e.g.,  $\Pi_d$  is the set of 3-colorable  $d$ -regular graphs [4]). To establish the upper bound in the subgraph model, observe that for any base graph  $G$  that has maximum degree  $d$ , the only subgraph of  $G$  that may be  $d$ -regular is  $G$  itself. Therefore, if the base graph  $G$  is not in  $\Pi_d$ , then the subgraph-tester can reject without performing any queries. If  $G \in \Pi_d$ , then the subgraph-tester simply tests whether the subgraph of  $G$  is  $G$  itself (by performing  $O(1/\epsilon)$  queries). ◀

The proof of Theorem 1.8 begs the question of whether the theorem holds also for downward-monotone properties, and more generally, which properties  $\Pi_d$  can be tested using  $O(1/\epsilon)$  queries in the subgraph model *w.r.t. every base graph of maximum degree  $d$* ? Alternatively, one may reverse the order of quantifiers and ask whether there exists a graph property  $\Pi$  that satisfies the conclusion of Theorem 1.8 *for any constant  $d$* .

### 1.3 Techniques

Some of the testers (algorithms) presented in this paper (e.g., Theorems 1.3 and 1.7) are based on structural properties of the base graph. In some cases (e.g., Theorem 1.3) these structural properties, which are inherited by the subgraphs, make the testing task (in the subgraph model) easier than in the BDG model. The proofs of the lower bounds constitute the more technically challenging part of the paper. Typically, the challenge is emulating lower bounds obtained for other testing models on the subgraph testing model. The brief overviews, especially those referring to the lower bounds, are merely intended to give a flavor of the techniques (and are not supposed to convince the reader of the validity of the claims).

### 1.3.1 Algorithms

The tester used in proving Theorem 1.2 is a simple emulation of the BDG-model tester by the subgraph tester, and its analysis is based on the observation that the distance between a graph  $G'$  and a downward-monotone graph property  $\Pi$  equals the number of edges that should be omitted from  $G'$  in order to place the resulting graph in  $\Pi$ . Proposition 1.6 is also proved by a simple emulation of the BDG-model tester, but the analysis of the resulting tester relies on special features of connectivity (and does not extend to 2-connectivity; see Problem 1.9).

The proofs of Theorems 1.3 and 1.7 are more interesting. In both cases we reduce testing subgraphs of the base graph  $G$  to testing subgraphs of a fixed subgraph  $G'$  of  $G$  such that  $G'$  is close to  $G$  and testing subgraphs of  $G'$  is (or seems) relatively easier. Such a reduction is valid since the property that we test is downward-monotone, and the subgraph  $G'$  is found without making any queries.

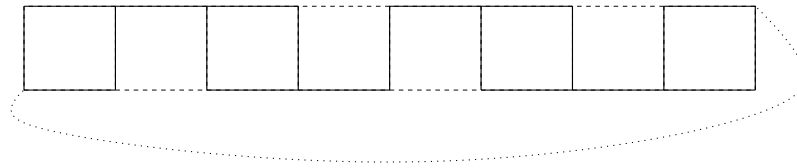
In the proof of Theorem 1.3 the fixed subgraph  $G'$  consists of small connected components. Hence, in the special case of Theorem 1.3 (i.e., properties that are determined by their connected components), it suffices to test that the subgraphs induced by the connected components of the base graph have the relevant property. In the general case, we follow Newman and Sohler [29] in estimating the frequency of the various graphs that are seen in these induced subgraphs. We stress that, unlike in [29], we do not use a *partition oracle of the tested graph* (which may be implemented based on standard queries (following Hassidim et al. [22])), but rather determine such a *partition of the base graph* (without making any queries).

Theorem 1.7 is proved by applying a recursive decomposition of the base graph using constant-size separating sets. Essentially, in addition to checking the local neighborhood of random vertices, we also check the local neighborhoods of the vertices in the separating sets that correspond to the path in the recursion tree (i.e., the tree of recursive decomposition) that isolates the chosen vertices. Actually, we check that all these local neighborhoods are consistent with some subgraph that has the property. These additional checks are used in the analysis in order to establish the consistency of the various local neighborhoods (i.e., not only those examined in the same execution).

We highlight the fact that the foregoing testers are non-adaptive. This is remarkable, because the corresponding testers for the BDG model (which in some cases are actually emulated by our testers) are inherently adaptive. However, these “BDG model testers” utilize their adaptivity only for conducting local searches in the input graph, whereas in the subgraph testing model the input is a subgraph of a fixed (or *a priori* known) graph, and so the queries that support these local searches can be determined non-adaptively.

### 1.3.2 Lower bounds

The lower bound on testing 3-colorability of a subgraph (asserted in Part 2 of Theorem 1.4) is established by combining the query complexity lower bound of [2] with a variant of the standard reduction of 3SAT to 3COL (cf. [12, Prop. 2.27]). Recall that Ben-Sasson et al. [2] prove the existence of (sparse) 3CNF formulae for which testing satisfiability of a given assignment requires linear query complexity. We reduce this testing problem, which refers to an explicitly given 3CNF formula (viewed as a massive parameter), to testing 3-colorability of a subgraph of a base graph. That is, we view the NP-reduction of 3SAT to 3COL as a mapping of a parameter (i.e., 3CNF formula) of one testing problem to a parameter (i.e., base graph) of another testing problem. In addition, we establish a one-to-one correspondence between



■ **Figure 2** A subgraph of the 2-by-8 grid that misses 4 edges. The subgraph is marked by solid lines, the missing edges by dashed lines, and an external edge that makes this subgraph 2-connected is dotted.

the bits of the assignment and part of the edges in the base graph, while considering only subgraphs that contain all other edges of the base graph (i.e., those not in correspondence to the bits of the assignment).

The proof of Item 1 of Theorem 1.4 (which refers to testing 2-colorability of a subgraph) is more complicated. The basic idea is to emulate the lower bound on bipartite testing established in the BDG model [17]. The obvious question is what should be the base graph. It is natural to pick a base graph that allows an embedding of any bounded-degree graph in it such that edges of the embedded graph are mapped to short vertex-disjoint paths. Furthermore, the mapping of edges to paths should be determined in a local manner. We use a base graph that is related to a routing network of logarithmic depth, and employ (randomized) oblivious routing on it. This allows us to map bipartite graphs (of the BDG model) to bipartite subgraphs of the base graph, while mapping graphs that are far from bipartite (in the BDG model) to subgraphs that are far from bipartite (in the subgraph testing model). The actual analysis of this construction is quite complicated (as evident from the length of [19, Sec. 3.1]), because we have to *locally emulate* a subgraph of the base graph (by making few queries to the input graph in the BDG model).

The proof of Theorem 1.5 uses a reduction from testing Eulerian orientations of cyclic grids in the orientation model. As discussed in Section 1.5.2, the orientation model (presented by Halevy et al. [21]) is related but different from the subgraph testing model. Our reduction maps the (cyclic) grid, used in the lower bound of Fischer et al. [11], to a base graph that looks like such a grid, except that edges are replaced by small gadgets. The orientations of edges in the orientation model are mapped to choices of subgraphs of the corresponding gadgets. In this case, it is easy to locally emulate a subgraph of the base graph by making queries to the orientation oracle, and the claimed  $\Omega(\log \log n)$  lower bound follows (from the analogous lower bound of [11]). On the other hand, the property at the image of the reduction is local, and so it is testable within  $\text{poly}(1/\epsilon)$  queries in the BDG model.

## 1.4 Open problems

Moving from the BDG model to the subgraph testing model makes the testing task potentially easier, since the subgraph tester knows *a priori* the possible locations of edges. But, when dealing with properties that are not downward-monotone, there is an opposite effect that arises from the fact that the distance to the set of subgraphs (of  $G$ ) that have graph property  $\Pi$  may be much bigger than the distance to the set of (bounded-degree) graphs that have property  $\Pi$ . This may require the subgraph tester to reject the input (since its distance to  $\Pi \cap \mathcal{F}_G$  is large), whereas the BDG model tester may be allowed to accept the input (since its distance  $\Pi$  is small). This difficulty is reflected in the following open problems.

► **Problem 1.9** (testing 2-connectivity of subgraphs). *Is the query complexity of testing 2-edge-connectivity in the subgraph testing model independent of the size of the graph? What about  $c$ -edge-connectivity for any constant  $c \in \mathbb{N}$ ?*



Recall that  $c$ -connectivity is testable in the BDG model within complexity that depends only on the proximity parameter [17]. We note that a straightforward emulation of the BDG-model tester (for 2-connectivity) calls for trying to find a small 2-connected component that has a cut of size at most 1 to the rest of the graph. But this approach fails when considering a base graph that is a 2-by- $n$  grid (since, as illustrated in Figure 2, any subgraph that misses at most one horizontal edge of each vertex (of degree 4) is  $O(1/n)$ -close to being 2-connected but may be far from any 2-connected subgraph of the 2-by- $n$  grid).

The straightforward emulation of the BDG-model tester also fails for testing whether a subgraph of the  $n$ -cycle is a perfect matching (i.e., is 1-regular), but a tester that considers the locations of edges in the subgraph does work (we discuss this shortly at the very end of [19, Sec. 4]). Testers of complexity that does not depend on the graph size do exist for this property when the base graph is a tree (since each tree has at most one perfect matching)<sup>9</sup>, but we do not know if they exist when the graph is outerplanar.

► **Problem 1.10** (testing whether the subgraph is a perfect matching). *What is the complexity of testing 1-regularity when the base graph is outerplanar? What about the case that the base graph is planar (e.g., a grid)? And what about testing degree-regularity?*

Note that  $c$ -connectivity, degree-regularity, and Eulerianity are the only properties covered in [14, Chap. 9] that are not downward-monotone. Also note that [19, Prop. 4.2] refers to the complexity of testing the Eulerian property for a base graph that is a grid, and it is clear that the underlying ideas apply to base graphs of “similar structure” (as arising in the proof of [19, Prop. 4.2]). But what about going beyond that?

► **Problem 1.11** (testing whether the subgraph is Eulerian). *What is the complexity of testing the Eulerian property in any base graph of bounded degree?*

The foregoing problems are all rooted in the difficulties that are introduced by the fact that distances under the subgraph model may be significantly larger than under the BDG model, which makes the task of the tester potentially harder. On the other hand, the fact that the base graph is known to the tester makes its task potentially easier. Recalling that only the latter effect is relevant in the case of downward-monotone properties, begs the following question.

► **Problem 1.12** (a property that is always easier in the subgraph model). *Does there exist a downward-monotone graph property  $\Pi$  such that testing  $\Pi$  in the bounded-degree model has higher query complexity than testing  $\Pi$  in the subgraph model w.r.t. every base graph of bounded-degree?*

Recall that Theorem 1.8 refers to an upward-monotone property (which depends on the degree bound).

The foregoing problems are aimed at concretizing the abstract challenge of making better use of the knowledge of the base graph that is available to the tester. Although Theorem 1.4 indicates that this extra knowledge is not always helpful, other results point out to cases in which it is helpful. We would like to see more such cases and more substantial use of the knowledge of the base graph.

---

<sup>9</sup> This perfect matching is determined by a pruning process (started at the leaves), and the tester just checks that the subgraph equals this perfect matching (if it exists). Note that, also in this case, the tester does not emulate the BDG-model tester (which just samples vertices and checks their degree); such an emulation will fail poorly (even when the base graph is a path).

## 1.5 Related models

### 1.5.1 Testing under a promise

As mentioned earlier, testing graph properties under the promise that the tested graph has some (other) property was considered before (see discussion in [14, Sec. 12.2]). In fact, the bounded-degree graph model itself may be viewed as postulating such a promise. More conspicuous cases include the study of testing under the promise that the graph is hyperfinite [29] or more specifically planar [3], or with bounded tree-width [7]. In continuation to Theorem 1.2, we observe that testing downward-monotone graph properties in the subgraph model can be reduced to testing the same property under a promise that contains the base graph.

► **Theorem 1.13** (a generalization of Theorem 1.2). *Let  $\mathcal{G}$  and  $\Pi$  be downward-monotone graph properties such that  $\mathcal{G}$  contains graphs of degree at most  $d$ . Suppose that, when promised that the tested graph is in  $\mathcal{G}$ , the property  $\Pi$  is testable (in the bounded-degree graph model) with query complexity  $Q_{\mathcal{G}}(\cdot, \cdot)$ , where  $Q_{\mathcal{G}}$  is a function of the proximity parameter and (possibly) the size of the graph. Then, for every base graph  $G = ([n], E)$  in  $\mathcal{G}$ , testing whether a subgraph of  $G$  satisfies  $\Pi$  (with proximity parameter  $\epsilon$ ) can be done with query complexity  $d \cdot Q_{\mathcal{G}}(\epsilon', n)$ , where  $\epsilon' = \epsilon/d$ .*

Hence, results weaker than Theorem 1.3 may be obtained by combining Theorem 1.13 with the tester provided in [29] (see discussion in Section 2.2). Indeed, the improved results are due to the fact that in the subgraph model the tester is given the base graph for free. In the current case (of hyperfinite graphs), the tester does not need to query the tested graph in order to obtain a partition oracle of the tested graph; it may just use an adequate partition of the base graph. In general, a main challenge in the study of the subgraph model is in how to utilize the knowledge of the base graph in order to improve the complexity of testing.

### 1.5.2 The orientation model

A property testing model that is related to the subgraph model is the *orientation model*, which was introduced by Halevy et al. [21]. Similarly to the subgraph model, in the orientation model there is a fixed (undirected) base graph  $G = ([n], E)$ . However, the goal in the latter model is to test properties of *directed graphs* (digraphs) that are defined by *orientations* of the edges of  $G$ . That is, for each edge  $\{u, v\} \in E$ , either the edge is directed from  $u$  to  $v$ , or from  $v$  to  $u$ , and the algorithm may perform queries in order to obtain the orientation of edges of its choice. For a property  $\Pi$  of digraphs, the algorithm should distinguish (with probability at least  $2/3$ ) between the case that the tested orientation  $\vec{G}$  has the property  $\Pi$  and the case in which the orientation of more than  $\epsilon \cdot |E|$  edges should be flipped in order to obtain the property.

While the subgraph model and the orientation model are syntactically identical, the semantics are very different, as we explain next. Similarly to the subgraph model, an orientation  $\vec{G} = ([n], \vec{E})$  of  $G$  is defined by a function  $f : E \rightarrow \{0, 1\}$ . Here,  $f(e) = 1$  indicates that in  $\vec{G}$  the edge  $e$  is directed from its smaller (id) endpoint to its larger endpoint. Querying the orientation of an edge hence corresponds to querying  $f$ , and distance between two functions  $f$  and  $f'$  (representing two different digraphs) is simply the Hamming distance between the functions.

The fundamental difference in the semantic between the two models is reflected in the fact that natural properties of digraphs in the orientation model do not correspond to nature properties of graphs in the subgraph testing model, and vice versa. For example, the functions

$f$  that define Eulerian orientations of an undirected graph  $G = ([n], E)$  (as described above) do not necessarily define subgraphs of  $G$  (i.e., in which  $f(e) = 1$  indicates that  $e$  belongs to the subgraph) that are Eulerian. Hence, natural properties in one model do not necessarily translate to natural properties in the other model. Still, it may be possible to emulate or reduce testing properties in one model to testing properties in the other model, as we do in the proof of Theorem 1.5.

## 2 Algorithms

In this section we prove Theorems 1.2, 1.3, and 1.7 (as well as Propositions 1.6 and 2.1). These results refer to different types of base graphs and different classes of properties. We have organized them according to the type of the base graph. Recall that  $G$  is assumed to have no isolated vertices, so that  $|E| \geq n/2$ .

### 2.1 General bounded-degree base graphs

In this section  $d \geq 2$  is a fixed constant, and the base graph  $G$  is an arbitrary graph in which each vertex has degree at most  $d$  (and at least 1).

#### 2.1.1 Testing downward-monotone properties

We first consider any graph property  $\Pi$  that is preserved under edge omission. Such properties are called *downward-monotone* or *downwards monotone*. We prove Theorem 1.2, which asserts that *for every graph  $G = ([n], E)$  of degree at most  $d$  and any downward-monotone graph property  $\Pi$ , testing  $\Pi \cap \mathcal{F}_G$  in the subgraph model (w.r.t. the base graph  $G$ ) is not harder than testing  $\Pi$  in the bounded-degree graph (BDG) model.*

**Proof of Theorem 1.2.** Given oracle access to  $f : E \rightarrow \{0, 1\}$ , the subgraph tester invokes the tester of the BDG model, and emulates an incidence oracle for the subgraph of  $G$  represented by  $f$  in the natural manner. That is, the query  $(v, i) \in [n] \times [d]$  is answered with the  $i^{\text{th}}$  vertex in the set  $\Gamma_f(v) = \{u : \{u, v\} \in E \ \& \ f(u, v) = 1\}$ , where vertices are ordered arbitrarily (e.g., by lexicographic order), and the answer is  $\perp$  if  $|\Gamma_f(v)| < i$ . This means that each query  $(v, i)$  of the BDG model tester, denoted  $T$ , is answered by first retrieving  $\Gamma_f(v)$ , which in turn amounts to making at most  $d$  queries to  $f$  (i.e., querying all edges incident to  $v$  in  $G$ ). Hence, the subgraph tester emulates the execution of  $T$  on the graph  $G^f = ([n], \{e \in E : f(e) = 1\})$ .

In the analysis, downward monotonicity is used to associate distance from  $\Pi$  in each of the two models with the number of edges that should be omitted from the subgraph. Specifically, in both cases, the distance from the property reflects the number of edges that should be omitted in order to make the graph satisfy the property (because adding edges never decreases that distance). Specifically, if  $f \in \Pi \cap \mathcal{F}_G$ , then  $G^f \in \Pi$ , and  $T$  accepts (with probability at least  $2/3$  in general, and with probability 1 if  $T$  has one-sided error). On the other hand, if  $f : E \rightarrow \{0, 1\}$  is  $\epsilon$ -far from  $\Pi \cap \mathcal{F}_G$ , then (by downward-monotonicity of  $\Pi$ ) any graph in  $\Pi$  that is closest to  $G^f$  must be a subgraph of  $G^f$  (i.e., is in  $\Pi \cap \mathcal{F}_G$  and so differs from  $G^f$  on more than  $\epsilon \cdot |E|$  edges). It follows that  $G^f$  is  $\epsilon'$ -far from  $\Pi$  for  $\epsilon' = \frac{\epsilon \cdot |E|}{dn/2} \geq \frac{\epsilon}{d}$ .  $\blacktriangleleft$

**Proof of Theorem 1.13.** The proof is identical to the proof of Theorem 1.2, except that here we rely on the hypothesis that  $\mathcal{G}$  is downward-monotone.  $\blacktriangleleft$

### 2.1.2 Testing monotone properties

Theorem 1.2 does not apply to monotone properties. Still, several such properties are quite easy to test in the subgraph testing model. One simple example is the property of having a specified minimal degree.

► **Proposition 2.1** (testing minimal degree in the subgraph model). *For  $d' \geq 1$ , testing whether all vertices in the subgraph have degree at least  $d'$  can be done in time  $O(d/\epsilon)$ .*

**Proof.** If  $d'$  is bigger than the minimum degree of the base graph  $G = ([n], E)$ , then the tester rejects without performing any queries. Otherwise, the tester selects  $\Theta(1/\epsilon)$  vertices, uniformly at random, and computes their degrees in the tested subgraph  $G^f$ , by querying all their incident edges in  $G$ . The tester accepts if and only if all sampled vertices have degree at least  $d'$ .

Hence, the tester makes  $O(d/\epsilon)$  queries, and always accepts subgraphs that have the property. To prove that it rejects subgraphs that are  $\epsilon$ -far from having the property with probability at least  $2/3$ , we establish the contrapositive statement. Consider a graph  $G^f$  that is accepted with probability at least  $1/3$ . This implies that the number of vertices in  $G^f$  whose degree is smaller than  $d'$  is at most  $(\epsilon/2) \cdot n$ . Since in  $G$  every vertex has degree at least  $d'$ , it is possible to add edges to  $G^f$  in order to obtain a subgraph that has the property, whereas the number of required added edges is at most  $(\epsilon n/2) \cdot d' \leq \epsilon \cdot |E|$ . ◀

**Proof of Proposition 1.6.** We now turn to the proof of Proposition 1.6, which asserts a  $\text{poly}(d/\epsilon)$ -time tester for connectivity in the subgraph model. If the base graph  $G = ([n], E)$  is not connected, then testing is trivial (since all subgraphs of  $G$  are disconnected). Otherwise (i.e., the base graph  $G$  is connected), connectivity of the input  $f \in \mathcal{F}_G$  can be tested by emulating the tester used for the BDG model [17]. This tester samples vertices and explores their local neighborhood in search of small connected components.

The analysis is even simpler than the original (bounded-degree) one since we can add edges without worrying about the degree bound (similarly to the analysis of testing connectivity in the sparse (unbounded-degree) model [30]). Specifically, we use the fact that if  $f$  represents a subgraph with  $t$  connected components, then by modifying  $f$  at one entry we can obtain a function that represents a subgraph with  $t - 1$  connected components. (This relies on the fact that  $G$  must contain edges between the connected components of  $G^f$ .) ◀

As noted in the introduction (see Section 1.4), the argument does not extend to 2-connectivity. The reason is that in that case the known tester for the BDG model [17] does not search for arbitrary 2-connected components but rather for 2-connected components that are connected to the rest of the graph by at most one edge. The problem with that tester is that its analysis requires the ability to add edges between any given pair of such 2-connected components, whereas we can only add edges that exist in the base graph.

## 2.2 Hyperfinite base graphs

A graph  $G = ([n], E)$  is said to have an  $(\epsilon, t)$ -partition if its vertex set can be partitioned into connected components of size at most  $t$  such that the number of edges between these components is at most  $\epsilon n$ .

Recall that a graph  $M$  is called a **minor** of a graph  $G$  if  $M$  is isomorphic to a graph that can be obtained by (zero or more) edge contractions on a subgraph of  $G$ . A graph  $G$  is  $M$ -minor free if  $M$  is not a minor of  $G$ . If  $G$  has degree at most  $d$  and is minor-free (i.e.,  $G$  is  $M$ -minor free for some fixed subgraph  $M$ ), then it has an  $(\epsilon, O((d/\epsilon)^2))$ -partition, for every  $\epsilon > 0$  (the size of  $M$  is “hidden” in the  $O(\cdot)$  notation – see [1, 22]).

More generally, Theorem 1.3 refers to any family of hyperfinite graphs, where a family of graph  $\mathcal{G}$  is hyperfinite if there exists a function  $t : (0, 1) \rightarrow \mathbb{N}$  such that, for every  $\epsilon > 0$ , every graph in the family has an  $(\epsilon, t(\epsilon))$ -partition. We shall first prove the second clause in the theorem, which refers to downward-monotone properties that are additive (i.e., determined by the connected components of the graph).

**A special case of interest.** We say that a graph property  $\Pi$  is additive if it holds that a graph is in  $\Pi$  if and only if all its connected components are in  $\Pi$ . We comment that not every downward-monotone graph property is additive. For example, consider the graph property  $\Pi$  that consists of all graphs that either constitute of a single (Hamiltonian) cycle or consist of a collection of isolated paths and vertices. Note that  $\Pi$  is closed under omission of edges and vertices, but a graph that consists of several isolated cycles is not in  $\Pi$  (i.e.,  $\Pi$  is not additive).<sup>10</sup>

► **Proposition 2.2** (testing downward-monotone properties that are additive). *Let  $\Pi \neq \emptyset$  be a downward-monotone graph property that is additive. Let  $G = ([n], E)$  be a graph of maximum degree  $d$ , and  $t : (0, 1) \rightarrow \mathbb{N}$  be such that, for every  $\epsilon > 0$ , the graph  $G$  has an  $(\epsilon, t(\epsilon))$ -partition. Then, testing whether a subgraph of  $G$  is in  $\Pi$  can be done by performing  $O(d^2 \cdot t(\epsilon/4)/\epsilon)$  queries. Furthermore, the tester is non-adaptive and has one-sided error.*

In particular, Proposition 2.2 implies that, for every fixed graph  $M$ , testing bipartiteness of a subgraph of  $G$ , when  $G$  is  $M$ -minor free, can be done in  $\text{poly}(d/\epsilon)$ -time, when given an  $(\epsilon/4, O((d/\epsilon)^2))$ -partition of  $G$ .<sup>11</sup> This is much more efficient than testing bipartiteness in the bounded-degree model, for which the query complexity is  $\Omega(\sqrt{n})$  [17]. It is also more efficient than testing bipartiteness of bounded-degree graphs under the promise that the graph is minor-free, let alone under the weaker promise that the graph is  $t$ -hyperfinite. Indeed, under these promises, the tester may implement an  $(\epsilon/4, t(\epsilon/4))$ -partition oracle of the tested subgraph, but such an implementation requires more than  $\text{poly}(t(\epsilon/4))$  queries. Specifically, in the special case of minor-free graphs the best implementation known uses  $O(d/\epsilon)^{O(\log(1/\epsilon))}$  queries [26], whereas in the general ( $t$ -hyperfinite) case the best implementation known uses  $\exp(d^{O(t(\text{poly}(1/\epsilon)))})$  queries [22],

**Proof Sketch.** Let  $(C_1, \dots, C_r)$  be an  $(\epsilon/4, t(\epsilon/4))$ -partition of  $G$ . Given query access to  $f : E \rightarrow \{0, 1\}$ , which represents the subgraph  $G^f = ([n], \{e \in E : f(e) = 1\})$ , we select at random  $\Theta(d/\epsilon)$  vertices, and for each selected vertex  $v$  we inspect all edges in the subgraph of  $G = ([n], E)$  induced by the part  $C_i$  that contains  $v$  (i.e., we query all pairs  $(u, w) \in E \cap (C_i \times C_i)$ ). We accept if and only if all the retrieved subgraphs are in  $\Pi$ ; that is, we accept if and only if for each inspected  $C_i$  it holds that the subgraph of  $G^f$  induced by  $C_i$  is in  $\Pi$ .

Using the fact that  $\Pi$  is preserved by omission of edges (and omission of connected components), we observe that if  $G^f$  is in  $\Pi$ , then so are the subgraphs of  $G^f$  induced by the  $C_i$ 's. Hence, our tester accepts  $G^f \in \Pi$  with probability 1. On the other hand, if  $G^f$  is  $\epsilon$ -far from  $\Pi$ , then the subgraph of  $G^f$  obtained by omitting all edges between the  $C_i$ 's is  $(\epsilon/2)$ -far from  $\Pi$  (since  $(\epsilon/4)n \leq (\epsilon/2)|E|$ ). Denoting the latter subgraph by  $\hat{G}^f$ , we claim that at least  $(\epsilon/4)n/d$  of its vertices reside in connected components that are not in  $\Pi$ , and it follows that the tester rejects  $G^f$  with high probability (since each connected component is contained in one of the  $C_i$ 's).

<sup>10</sup> Indeed, if a graph is in (this)  $\Pi$ , then all its connected are in  $\Pi$ , but the converse does not hold.

<sup>11</sup> Such a partition can be found in polynomial-time [1].

The foregoing claim is proved by relying on the hypothesis that  $\Pi$  is additive. Specifically, if less than  $(\epsilon/4)n/d$  vertices reside in connected components that are not in  $\Pi$ , then by omitting less than  $(\epsilon/4)n < (\epsilon/2)|E|$  edges we can make all connected components reside in  $\Pi$  (since  $\Pi$  contains the graph consisting of a single vertex). This implies that the graph consisting of these modified connected components is in  $\Pi$ , which in turn contradicts the hypothesis that  $\hat{G}^f$  is  $(\epsilon/2)$ -far from  $\Pi$ . ◀

**Greater generality at larger cost.** A more general result refers to graph properties  $\Pi$  that are only preserved under edge omission (and to hyperfinite base graphs  $G$ ). The cost of this generalization is an increase in the query complexity of the tester, as asserted next.

► **Proposition 2.3** (testing general downward-monotone properties). *Suppose that  $\Pi$  is a downward-monotone graph property and that, for some  $t : [0, 1] \rightarrow \mathbb{N}$  and every  $\epsilon > 0$ , the graph  $G = ([n], E)$  has an  $(\epsilon, t(\epsilon))$ -partition. Then, we can test whether a subgraph of  $G$  is in  $\Pi$  with query complexity  $O(d^2 \cdot \exp(t(\epsilon/4)^2)/\epsilon^2)$ .*

We mention that the exponential dependence on  $t$  of query complexity of the foregoing tester is unavoidable (in the general case of downward-monotone graph properties). Consider, for example, the case that the base graph is an  $\sqrt{n}$ -by- $\sqrt{n}$  grid augmented by diagonal edges in each small square, and the following downward-monotone property  $\Pi$ : A graph is in  $\Pi$  if there exists a  $k$  such that the graph consists of connected component that are each a  $k$ -by- $k$  grid augmented by some of the foregoing diagonal edges such that at most half of the possible patterns occur in these small grids. Now, on proximity parameter  $\epsilon > 0$ , consider the task of distinguishing the case that the subgraph consists of  $0.1/\epsilon$ -by- $0.1/\epsilon$  grids in which half of the possible patterns occur from the case in which all patterns occur. A lower bound that is a square root of the number of patterns follows from a birthday paradox argument (and a lower bound that is almost linear follows from [34, 33]).

**Proof Sketch.** By the premise of the proposition, for every  $\epsilon > 0$ , the base graph  $G$  has an  $(\epsilon/4, t(\epsilon/4))$ -partition. Let  $g \in \mathcal{F}_G$  denote the all-ones function, and let  $g'$  be  $\epsilon/2$ -close to  $g$  and describe a subgraph of  $G$  in which each connected component has size at most  $t(\epsilon/4)$ . Hence,  $G^{g'}$  is a subgraph of  $G$  that is obtained from  $G$  by removing the at most  $(\epsilon/4)n \leq (\epsilon/2)|E|$  edges between parts in the  $(\epsilon/4, t(\epsilon/4))$ -partition.

By the closure of  $\Pi$  to edge omissions, every function  $f \in \mathcal{F}_G \cap \Pi$  is  $0.5\epsilon$ -close to the function  $f' \in \mathcal{F}_G \cap \Pi$  such that  $f'(e) = f(e) \wedge g'(e)$ . Let  $\Pi'_G$  denote the set of graphs obtained in this way; that is,  $\Pi'_G = \{f \wedge g' : f \in \mathcal{F}_G \cap \Pi\}$ . Since  $\Pi$  is a graph property, it follows that  $\Pi'_G = \mathcal{F}_G \cap \Pi'$ , where  $\Pi'$  is the set of all graphs that are isomorphic to graphs that appear in  $\Pi'_G$ . Hence, the set  $\Pi'_G$  is closed under all automorphisms of the graph  $G$ .

Recalling that  $\Pi'_G$  and likewise  $\Pi'$  contain only graphs that consist of connected components of size at most  $t = t(\epsilon/4)$ , it follows  $\Pi'$  is characterized by the frequencies in which the various graphs of size at most  $t$  appear as connected components. Hence,  $f \in \mathcal{F}_G$  describes a graph in  $\Pi$  if and only if  $f' = f \wedge g'$  is in  $\mathcal{F}_G \cap \Pi'$ , where  $\Pi'$  is characterized in terms of the number of connected component that are isomorphic to each of the graphs with at most  $t(\epsilon/4)$  vertices (and contain no smaller connected components). It follows that testing with proximity parameter  $\epsilon$  whether subgraphs of  $G$  satisfy  $\Pi$  can be performed by estimating these numbers in the subgraph described by  $f \wedge g'$ , where  $f$  is the tested function. Lastly, we note that estimating the frequencies in which the various  $t(\epsilon/4)$ -vertex graphs appear as connected components can be done using  $O(d^2 \cdot \exp(t(\epsilon/4)^2)/\epsilon^2)$  queries, where  $\exp(t(\epsilon/4)^2)$  account for the number of  $t(\epsilon/4)$ -vertex graphs. ◀



### 2.3 Local properties and base graphs with small separators

Loosely speaking, a graph property is called *local* if satisfying it can be expressed as the conjunction of local conditions, where each local condition refers to a constant-distance neighborhood of one of the graph's vertices. A precise definition is given next.

► **Definition 2.4.** For a constant  $\ell \in \mathbb{N}$ , the  $\ell$ -neighborhood of a vertex  $v$  in a graph  $G$  is the subgraph of  $G$  induced by all vertices that are at distance at most  $\ell$  from  $v$ . A property  $\Pi$  of  $n$ -vertex graphs is called  $\ell$ -local if there exists a graph property  $\Pi'$  such that  $G$  is in  $\Pi$  if and only if the  $\ell$ -neighborhood of each vertex in  $G$  is in  $\Pi'$ . (Actually,  $\Pi'$  is a set of rooted graphs, where the root corresponds to the “center” of the  $\ell$ -neighborhood.)<sup>12</sup> A graph property  $\Pi = \bigcup_n \Pi_n$  is local if there exists a constant  $\ell$  such that  $\Pi_n$  is an  $\ell$ -local property of  $n$ -vertex graphs.

We mention that this definition coincides with [18, Def. 5.2], and that (in the bounded degree graph model) every graph property that has a proximity-oblivious tester of constant query complexity is local [18, Sec. 5].

For  $s : \mathbb{N} \rightarrow \mathbb{N}$  we say that a graph  $G = ([n], E)$  has separating sets of size  $s$  if for every set of vertices  $U \subseteq [n]$  there exists a subset  $S \subseteq U$  of at most  $s(|U|)$  vertices such that the subgraph of  $G$  induced by  $U \setminus S$  has no connected component of size greater than  $\frac{2}{3} \cdot |U|$ . For example, every tree has separating sets of size 1, every outerplanar graph has separating sets of size 2 [23, Lem. 3], and  $n$ -vertex planar graphs have separating sets of size  $O(\sqrt{n})$  [27].

► **Theorem 2.5** (Theorem 1.7, generalized). *Let  $\Pi$  be an  $\ell$ -local property and let  $s : \mathbb{N} \rightarrow \mathbb{N}$ . Suppose that the base graph  $G$  is of bounded degree  $d$  and has separators of size  $s$ . Then, testing whether a subgraph of  $G = ([n], E)$  has property  $\Pi$  can be done by performing  $O(\epsilon^{-1} s(n) \log n \cdot d^{\ell+1})$  queries. Furthermore, the tester is non-adaptive and has one-sided error.*

**Proof.** We consider a recursive decomposition of the graph  $G$ , obtained by applying the guaranteed separators, and a tree that corresponds to these applications. Specifically, the root of the tree corresponds to the separating set, denoted  $S_\lambda$ , that disconnects the graph  $G_\lambda \stackrel{\text{def}}{=} G$ . Collecting the resulting connected components into two subgraphs, each containing at most two-thirds of  $G$ 's vertices, we proceed to obtain separating sets, denoted  $S_0$  and  $S_1$ , for each of these two subgraphs, denoted  $G_0$  and  $G_1$ , respectively. In general, an internal node in the tree is labeled by a string  $\alpha$  and corresponds to the subgraph  $G_\alpha$  as well as to a separating set  $S_\alpha$  for  $G_\alpha$ . The children of this node correspond to subgraphs  $G_{\alpha 0}$  and  $G_{\alpha 1}$  that result from removing  $S_\alpha$  from  $G_\alpha$  (where the number of vertices in each of these subgraphs is at most two-thirds of the number of vertices in  $G_\alpha$ ). When the subgraph reaches some constant size, the process stops. Hence, the leaves of the tree correspond to subgraphs of constant size. For a leaf labeled by  $\alpha$ , we let  $S_\alpha$  be the set of vertices of the subgraph  $G_\alpha$ .

For the sake of clarity, we reserve the term ‘node’ for nodes in the tree (describing the recursive decomposition), and the term ‘vertex’ for the vertices of  $G$ . We shall never talk of edges of the (rooted) tree, but only of the descendance and ancestry relations induced by it. Recall that each node in the tree is associated with a set of vertices of  $G$ , and note that these sets form a partition of the vertex set of  $G$ . We say that vertex  $v$  resides in a node labeled by  $\alpha$  if  $v \in S_\alpha$ . Observe that edges of the graph  $G$  can connect vertices that reside in

<sup>12</sup>Marking the root is important only in case that the center of the graph of radius  $\ell$  cannot be uniquely determined.



the same node and vertices that reside in nodes that are in an ancestry relation, but *cannot* connect vertices that reside in nodes that are not in an ancestry relation (equiv., reside in nodes  $\alpha'0\alpha''$  and  $\alpha'1\alpha'''$  for any  $\alpha', \alpha'', \alpha''' \in \{0, 1\}^*$ ).

We are now ready to describe the tester for  $\Pi$ , which is an  $\ell$ -local property for some constant  $\ell \in \mathbb{N}$ . Given a fixed based graph  $G = ([n], E)$  and oracle access to a subgraph represented by  $f : E \rightarrow \{0, 1\}$ , the tester repeats the following procedure  $\Theta(d/\epsilon)$  times, where if no invocation of the procedure causes rejection, then it accepts.

1. Uniformly select a vertex that resides in one of the leaves of the decomposition tree. (Note that a constant fraction of the vertices of  $G$  resides in leaves of the tree.)
2. For each vertex  $v$  of  $G$  that resides in a node on the path from the selected leaf to the root (including both the leaf and the root), explore the  $\ell$ -neighborhood of  $v$  in  $G$  (i.e., query  $f$  on each of the edges in that neighborhood).
3. If the subgraph discovered in the previous step is not consistent with any  $n$ -vertex subgraph of  $G$  that has property  $\Pi$ , then reject.

Note that the aforementioned discovered subgraph includes not only the explored edges but also indication that certain edges do not exist in the subgraph (i.e., the latter include all non-edges of  $G$  as well as some edges of  $G$  that were queried by the procedure and answered by the value 0).

The query complexity of this procedure is  $O(s(n) \log n \cdot d^\ell)$ , where  $d$  is the degree-bound of  $G$ . Clearly, the tester always accepts subgraphs of  $G$  that have the property  $\Pi$ . It remains to show that if the subgraph is  $\epsilon$ -far from  $\Pi$ , then the probability that a single invocation of the procedure causes rejection is  $\Omega(\epsilon/d)$ .

We establish the contrapositive statement. Suppose that the foregoing procedure rejects with probability  $\rho < 1$ . We show that it suffices to modify an  $O(\rho \cdot d)$  fraction of the edges in  $G$  in order to obtain a graph that satisfies  $\Pi$ . We say that a leaf of the tree is **good** if the procedure does not reject when it selects a vertex that resides in this leaf. We say that an internal node of the tree is **good** if it appears on the path from some good leaf to the root. Note that  $\rho < 1$  implies that there exist good leaves, and hence the root of the tree is good. More generally, if a node is good, then all its ancestors are good. Also note that each vertex that resides in a good node has an  $\ell$ -neighborhood in  $G^f$  that satisfies the local condition (i.e., the  $\ell$ -neighborhood is in  $\Pi'$ ), where recall that  $G^f$  denotes the subgraph of  $G$  defined by  $f$ .

Hence, we only need to modify the neighborhoods of vertices residing in bad nodes, and we should do so without harming the neighborhoods of vertices that reside in good nodes. But before explaining how this is done, we note that the number of vertices that reside in internal nodes belonging to the subtree rooted in node  $\alpha$  is only a constant factor larger than the number of vertices that reside in the leaves of this subtree. On the other hand, considering the set of bad nodes that have good parents, we note that  $\rho$  equals the fraction of vertices that reside in leaves of the subtrees rooted at these bad nodes.

Consider an arbitrary bad node, denoted  $\alpha\sigma$ , that has a good parent, denoted  $\alpha$ . Then, the  $\ell$ -neighborhoods of the vertices residing in node  $\alpha$  satisfy the local condition (in the subgraph  $G^f$ ). We claim that the  $\ell$ -neighborhoods of vertices in  $G_{\alpha\sigma}$  can be modified so that they satisfy the local conditions as well without modifying the  $\ell$ -neighborhoods of any vertex that resides in a good node. To verify this claim observe the intersection of the  $\ell$ -neighborhoods of vertices in  $G_{\alpha\sigma}$  and the  $\ell$ -neighborhoods of vertices that reside in good nodes is contained in the intersection of the  $\ell$ -neighborhoods of vertices in  $G_{\alpha\sigma}$  and the  $\ell$ -neighborhoods of vertices that reside either in node  $\alpha$  or in one of its ancestors. The reasoning is that if vertex  $v$  in  $G_{\alpha\sigma}$  is adjacent in  $G$  to a vertex  $u$ , then either  $u$  is in  $G_{\alpha\sigma}$  or  $u$  is in  $S_{\alpha'}$  such that  $\alpha'$  is a (not necessarily proper) prefix of  $\alpha$ .

Recall that by Item 3 of the procedure (based on which the notion of good node is defined) the fact that node  $\alpha$  is good, implies that the  $\ell$ -neighborhoods of vertices in  $G_{\alpha\sigma}$  can be modified to satisfy  $\Pi'$  in a manner that is consistent with the  $\ell$ -neighborhoods of all vertices that reside in node  $\alpha$  and its ancestors, and so with the  $\ell$ -neighborhoods of all vertices that reside in good nodes. It follows that by modifying  $f$  on  $G_{\alpha\sigma}$ , while maintaining the  $\ell$ -neighborhoods of vertices in  $S_\alpha$  (as well as  $S_{\alpha'}$  for each  $\alpha'$  that is an ancestor of  $\alpha$ ) intact, we can “fix” the  $\ell$ -local neighborhood of all vertices in  $G_{\alpha\sigma}$ .

The foregoing process modifies  $f$  into a function that describes a subgraph of  $G$  that is in  $\Pi$ , while modifying  $O(\rho \cdot d \cdot n) = O(\rho \cdot d \cdot |E|)$  edges. The theorem follows. ◀

---

## References

- 1 N. Alon, P.D. Seymour, and R. Thomas. A separator theorem for graphs with an excluded minor and its applications. In *Proceedings of the Twenty-Second Annual ACM Symposium on Theory of Computing (STOC)*, pages 293–299, 1990.
- 2 E. Ben-Sasson, P. Harsha, and S. Raskhodnikova. 3CNF properties are hard to test. *SIAM Journal on Computing*, 35(1):1–21, 2005.
- 3 I. Benjamini, O. Schramm, and A. Shapira. Every minor-closed property of sparse graphs is testable. *Advances in mathematics*, 223:2200–2218, 2010.
- 4 A. Bogdanov, K. Obata, and L. Trevisan. A lower bound for testing 3-colorability in bounded-degree graphs. In *Proceedings of the Forty-Third Annual Symposium on Foundations of Computer Science (FOCS)*, pages 93–102, 2002.
- 5 A. Czumaj, M. Monemizadeh, K. Onak, and C. Sohler. Planar Graphs: Random Walks and Bipartiteness Testing. In *Proceedings of the Forty-Third Annual Symposium on Foundations of Computer Science (FOCS)*, pages 423–432, 2011.
- 6 A. Czumaj, A. Shapira, and C. Sohler. Testing Hereditary Properties of Nonexpanding Bounded-Degree Graphs. *SIAM Journal on Computing*, 38(6):2499–2510, 2009.
- 7 A. Edelman, A. Hassidim, H. N. Nguyen, and K. Onak. An Efficient Partitioning Oracle for Bounded-Treewidth Graphs. In *Proceedings of the Fifteenth International Workshop on Randomization and Computation (RANDOM)*, pages 530–541, 2011.
- 8 G. Elek. The combinatorial cost. *ArXiv Mathematics e-prints*, 2006. arXiv:math/0608474.
- 9 G. Elek. Parameter testing with bounded degree graphs of subexponential growth. *Random Structures and Algorithms*, 37:248–270, 2010.
- 10 P. Erdos and A. Renyi. Asymmetric graphs. *Acta Mathematica Hungarica*, 14(3):295–315, 1963.
- 11 E. Fischer, O. Lachish, A. Matsliah, I. Newman, and O. Yahalom. On the query complexity of testing orientations for being Eulerian. *ACM Transactions on Algorithms*, 8(2):15:1–15:41, 2012.
- 12 O. Goldreich. *Computational Complexity: A Conceptual Perspective*. Cambridge University Press, 2008.
- 13 O. Goldreich, editor. *Property Testing: Current Research and Surveys*. Springer, 2010. LNCS 6390.
- 14 O. Goldreich. *Introduction to Property Testing*. Cambridge University Press, 2017.
- 15 O. Goldreich, S. Goldwasser, and D. Ron. Property Testing and its Connection to Learning and Approximation. *Journal of the ACM*, 45(4):653–750, 1998.
- 16 O. Goldreich and D. Ron. A sublinear bipartite tester for bounded-degree graphs. *Combinatorica*, 19(3):335–373, 1999.
- 17 O. Goldreich and D. Ron. Property Testing in Bounded Degree Graphs. *Algorithmica*, 32(2):302–343, 2002.

- 18 O. Goldreich and D. Ron. On Proximity Oblivious Testing. *SIAM Journal on Computing*, 40(2):534–566, 2011.
- 19 O. Goldreich and D. Ron. The Subgraph Testing Model. *Electronic Colloquium on Computational Complexity (ECCC)*, 25:45, 2018.
- 20 M. Gonen and D. Ron. On the Benefit of Adaptivity in Property Testing of Dense Graphs. *Algorithmica*, 58(4):811–830, 2010. Special issue for APPROX-RANDOM 2007.
- 21 S. Halevy, O. Lachish, I. Newman, and D. Tsur. Testing Orientation Properties. *Electronic Colloquium on Computational Complexity (ECCC)*, 153, 2005.
- 22 A. Hassidim, J. Kelner, H. Nguyen, and K. Onak. Local Graph Partitions for Approximation and Testing. In *Proceedings of the Fiftieth Annual Symposium on Foundations of Computer Science (FOCS)*, pages 22–31, 2009.
- 23 L.S. Heath. Embedding outerplanar graphs in small books. *SIAM Journal on Algebraic and Discrete Methods*, 8(2):198–218, 1987.
- 24 T. Kaufman, M. Krivelevich, and D. Ron. Tight Bounds for Testing Bipartiteness in General Graphs. *SIAM Journal on Computing*, 33(6):1441–1483, 2004.
- 25 J.H. Kim, B. Sudakov, and V.H. Vu. On the asymmetry of random regular graphs and random graphs. *Random Structures and Algorithms*, 21(3-4):216–224, 2002.
- 26 R. Levi and D. Ron. A quasi-polynomial time partition oracle for graphs with an excluded minor. *ACM Transactions on Algorithms*, 11(3):24:1–24:13, 2015.
- 27 R.J. Lipton and R.E. Tarjan. A separator theorem for planar graphs. *SIAM Journal on Discrete Math*, 1979.
- 28 I. Newman. Property Testing of Massively Parametrized Problems – A Survey. In O. Goldreich, editor, *Property Testing - Current Research and Surveys*, pages 142–157. Springer, 2010. LNCS 6390.
- 29 I. Newman and C. Sohler. Every Property of Hyperfinite Graphs Is Testable. *SIAM Journal on Computing*, 42(3):1095–1112, 2013.
- 30 M. Parnas and D. Ron. Testing the Diameter of Graphs. *Random Structures and Algorithms*, 20(2):165–183, 2002.
- 31 D. Ron. Property Testing: A Learning Theory Perspective. *Foundations and Trends in Machine Learning*, 1(3):307–402, 2008.
- 32 D. Ron. Algorithmic and Analysis Techniques in Property Testing. *Foundations and Trends in Theoretical Computer Science*, 5(2):73–205, 2010.
- 33 G. Valiant. *Algorithmic Approaches to Statistical Questions*. PhD thesis, University of California at Berkeley, 2012.
- 34 G. Valiant and P. Valiant. Estimating the Unseen: Improved Estimators for Entropy and Other Properties. *Journal of the ACM*, 64(6):37:1–37:41, 2017.



# Adventures in Monotone Complexity and TFNP

**Mika Göös**<sup>1</sup>

Institute for Advanced Study, Princeton, NJ, USA  
mika@ias.edu

**Pritish Kamath**<sup>2</sup>

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

**Robert Robere**<sup>3</sup>

Simons Institute, Berkeley, CA, USA  
robere@cs.toronto.edu

**Dmitry Sokolov**<sup>4</sup>

KTH Royal Institute of Technology, Stockholm, Sweden  
sokolov.dmt@gmail.com

---

## Abstract

*Separations:* We introduce a monotone variant of XOR-SAT and show it has exponential monotone circuit complexity. Since XOR-SAT is in  $\text{NC}^2$ , this improves qualitatively on the monotone vs. non-monotone separation of Tardos (1988). We also show that monotone span programs over  $\mathbb{R}$  can be exponentially more powerful than over finite fields. These results can be interpreted as separating subclasses of TFNP in communication complexity.

*Characterizations:* We show that the communication (resp. query) analogue of PPA (subclass of TFNP) captures span programs over  $\mathbb{F}_2$  (resp. Nullstellensatz degree over  $\mathbb{F}_2$ ). Previously, it was known that communication FP captures formulas (Karchmer–Wigderson, 1988) and that communication PLS captures circuits (Razborov, 1995).

**2012 ACM Subject Classification** Theory of computation → Communication complexity, Theory of computation → Circuit complexity, Theory of computation → Proof complexity

**Keywords and phrases** TFNP, Monotone Complexity, Communication Complexity, Proof Complexity

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2019.38

**Related Version** A full version of the paper is available at [24], <https://ecc.weizmann.ac.il/report/2018/163/>.

**Acknowledgements** We thank Ankit Garg (who declined a co-authorship) for extensive discussions about monotone circuits. We also thank Thomas Watson and anonymous ITCS reviewers for comments.

---

<sup>1</sup> Work done while at Harvard University; supported by the Michael O. Rabin Postdoctoral Fellowship.

<sup>2</sup> Supported in parts by NSF grants CCF-1650733, CCF-1733808, and IIS-1741137.

<sup>3</sup> Work done while at University of Toronto; supported by NSERC.

<sup>4</sup> Supported by the Swedish Research Council grant 2016-00782, *Knut and Alice Wallenberg Foundation* grants KAW 2016.0066 and KAW 2016.0433.



## 1 Our Results

We study the complexity of *monotone* boolean functions  $f: \{0, 1\}^n \rightarrow \{0, 1\}$ , that is, functions satisfying  $f(x) \leq f(y)$  for every pair  $x \leq y$  (coordinate-wise). (An excellent introduction to monotone complexity is the textbook [36].) Our main results are new **separations** of monotone models of computation and **characterizations** of those models in the language of query/communication complexity. At the core of these results are two conceptual innovations.

1. We introduce a natural *monotone* encoding of the usual CSP satisfiability problem (Subsection 1.1). This definition unifies many other monotone functions considered in the literature.
2. We extend and make more explicit an intriguing connection between *circuit complexity* and *total NP search problems* (TFNP) via communication complexity. Several prior characterizations [37, 55] can be viewed in this light. This suggests a potentially useful organizational principle for circuit complexity measures; see Section 2 for our survey.

### 1.1 Monotone $\mathcal{C}$ -Sat

The basic conceptual insight in this work is a new simple definition: a *monotone encoding* of the usual constraint satisfaction problem (CSP). For any finite set of constraints  $\mathcal{C}$ , we introduce a monotone function  $\mathcal{C}$ -SAT. A general definition is given in Section 3, but for now, consider as an example the set  $\mathcal{C} = 3\text{XOR}$  of all ternary parity constraints

$$3\text{XOR} := \{ (v_1 \oplus v_2 \oplus v_3 = 0), (v_1 \oplus v_2 \oplus v_3 = 1) \}.$$

We define  $3\text{XOR-SAT}_n: \{0, 1\}^N \rightarrow \{0, 1\}$  over  $N := |\mathcal{C}|n^3 = 2n^3$  input bits as follows. An input  $x \in \{0, 1\}^N$  is interpreted as (the indicator vector of) a set of  $3\text{XOR}$  constraints over  $n$  boolean variables  $v_1, \dots, v_n$  (there are  $N$  possible constraints). We define  $3\text{XOR-SAT}_n(x) := 1$  iff the set  $x$  is *unsatisfiable*, that is, no boolean assignment to the  $v_i$  exists that satisfies all constraints in  $x$ . This is indeed a monotone function: if we flip any bit of  $x$  from 0 to 1, this means we are adding a new constraint to the instance, thereby making it even harder to satisfy.

**Prior work.** Our  $\mathcal{C}$ -SAT encoding generalizes several previously studied monotone functions.

**(NL)** Karchmer and Wigderson [37] (also [29, 49, 56] and textbooks [39, 36]) studied the NL-complete *st-connectivity* problem. This is equivalent to a  $\mathcal{C}$ -SAT problem with  $\mathcal{C}$  consisting of a binary implication ( $v_1 \rightarrow v_2$ ) and unit clauses ( $v_1$ ) and ( $\neg v_1$ ).

**(P)** Raz and McKenzie [52] (also [14, 15, 25, 18, 56, 48]) studied a certain P-complete *generation* problem. In hindsight, this is simply HORN-SAT, that is,  $\mathcal{C}$  consists of *Horn clauses*: clauses with at most one positive literal, such as ( $\neg v_1 \vee \neg v_2 \vee v_3$ ).

**(NP)** Göös and Pitassi [25] and Oliveira [45, §3] (also [47, 48]) studied the NP-complete (dual of) CNF-SAT problem, where  $\mathcal{C}$  consists of bounded-width clauses.

These prior works do not exhaust all interesting classes of  $\mathcal{C}$ , as is predicted by various classification theorems for CSPs [59, 20, 11, 63]. In this work, we focus on *linear* constraints over finite fields  $\mathbb{F}_p$  (for example,  $3\text{XOR-SAT}$  corresponding to  $\mathbb{F}_2$ ) and over the reals  $\mathbb{R}$ .

### 1.2 Separations

First, we show that  $3\text{XOR-SAT}_n$  cannot be computed efficiently with monotone circuits.

► **Theorem 1.**  $3\text{XOR-SAT}_n$  requires monotone circuits of size  $2^{n^{\Omega(1)}}$ .

This theorem stands in contrast to the fact that there exist fast parallel (non-monotone) algorithms for linear algebra [44]. In particular, 3XOR-SAT is in  $\text{NC}^2$ . Consequently, our result improves qualitatively on the monotone vs. non-monotone separation of Tardos [61] who exhibited a monotone function in  $\text{P}$  (computed by solving a semidefinite program) with exponential monotone circuit complexity. For further comparison, another famous candidate problem to witness a monotone vs. non-monotone separation is the *perfect matching* function: it is in  $\text{RNC}^2$  [40] while it is widely conjectured to have exponential monotone circuit complexity (a quasipolynomial lower bound was proved by Razborov [53]).

**Span programs.** The computational easiness of  $3\text{XOR-SAT}_n$  can be stated differently: it can be computed by a linear-size monotone  $\mathbb{F}_2$ -span program. Span programs are a model of computation introduced by Karchmer and Wigderson [38] (see also [36, §8] for exposition) with an extremely simple definition. An  $\mathbb{F}$ -span program, where  $\mathbb{F}$  is a field, is a matrix  $M \in \mathbb{F}^{m \times m'}$  each row of which is labeled by a literal,  $x_i$  or  $\neg x_i$ . We say that the program accepts an input  $x \in \{0, 1\}^n$  iff the rows of  $M$  whose labels are consistent with  $x$  (literals evaluating to true on  $x$ ) span the all-1 row vector. The *size* of a span program is its number of rows  $m$ . A span program is *monotone* if all its literals are positive; in this case the program computes a monotone function.

A corollary of Theorem 1 is that monotone  $\mathbb{F}_2$ -span programs cannot be simulated by monotone circuits without exponential blow-up in size. This improves on a separation of Babai, Gál, and Wigderson [3] who showed that monotone circuit complexity can be quasipolynomially larger than monotone  $\mathbb{F}_2$ -span program size.

Furthermore, Theorem 1 holds more generally over *any* field  $\mathbb{F}$ : an appropriately defined function  $3\text{LIN}(\mathbb{F})\text{-SAT}_n$  (ternary  $\mathbb{F}$ -linear constraints; see Section 3) is easy for monotone  $\mathbb{F}$ -span programs, but exponentially hard for monotone circuits. No such separation, even superpolynomial, was previously known for fields of characteristic other than 2.

This brings us to our second theorem.

► **Theorem 2.**  $3\text{LIN}(\mathbb{R})\text{-SAT}_n$  requires monotone  $\mathbb{F}_p$ -span programs of size  $2^{n^{\Omega(1)}}$  for any prime  $p$ .

In other words: monotone  $\mathbb{R}$ -span programs can be exponentially more powerful than monotone span programs over finite fields. This separation completes the picture for the relative powers of monotone span programs over distinct fields, since the remaining cases were exponentially separated by Pitassi and Robere [48].

Finally, our two results above yield a bonus result in proof complexity as a byproduct: the Nullstellensatz proof system (over any field) can be exponentially more powerful than the Cutting Planes proof system (see Subsection 4.2).

**Techniques.** The new lower bounds are applications of the lifting theorems for monotone circuits [23] and monotone span programs [48]. We show that, generically, if some unsatisfiable formula composed of  $\mathcal{C}$  constraints is hard to refute for the Resolution (resp. Nullstellensatz) proof system, then the  $\mathcal{C}$ -SAT problem is hard for monotone circuits (resp. span programs). Hence we can invoke (small modifications of) known Resolution and Nullstellensatz lower bounds [8, 10, 1]. The key conceptual innovation here is a reduction from unsatisfiable  $\mathcal{C}$ -CSPs (or their lifted versions) to the monotone Karchmer–Wigderson game for  $\mathcal{C}$ -SAT. This reduction is extremely slick, which we attribute to having finally found the “right” definition of  $\mathcal{C}$ -SAT.



### 1.3 Characterizations

There are two famous “top-down” characterizations of circuit models (both monotone and non-monotone variants) using the language of communication complexity; these characterizations are naturally related to communication analogues of subclasses of TFNP.

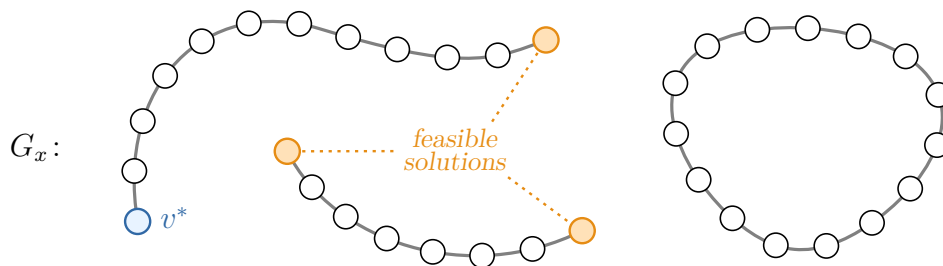
**(FP)** Karchmer and Wigderson [37] showed that the logarithm of the (monotone) formula complexity of a (monotone) function  $f: \{0, 1\}^n \rightarrow \{0, 1\}$  is equal, up to constant factors, to the communication complexity of the (monotone) *Karchmer–Wigderson game*:

$$\begin{array}{l} \text{Search problem } \mathbf{KW}(f) \\ \text{[resp. } \mathbf{KW}^+(f)\text{]} \end{array} = \begin{array}{l} \text{input: a pair } (x, y) \in f^{-1}(1) \times f^{-1}(0) \\ \text{output: an } i \in [n] \text{ with } x_i \neq y_i \text{ [resp. } x_i > y_i\text{]} \end{array}$$

We summarize this by saying that *the communication analogue of FP captures formulas*. Here  $\text{FP} \subseteq \text{TFNP}$  is the classical (Turing machine) class of total NP search problems efficiently solved by deterministic algorithms [42].

**(PLS)** Razborov [55] (see also [50, 60]) showed that the logarithm of the (monotone) circuit complexity of a function  $f: \{0, 1\}^n \rightarrow \{0, 1\}$  is equal, up to constant factors, to the least cost of a PLS-protocol solving the  $\text{KW}(f)$  (or  $\text{KW}^+(f)$ ) search problem. Here a PLS-protocol (Definition 14 in Appendix A) is a natural communication analogue of  $\text{PLS} \subseteq \text{TFNP}$  [35]. We summarize this by saying that *the communication analogue of PLS captures circuits*.

We contribute a third characterization of this type: *the communication analogue of PPA captures  $\mathbb{F}_2$ -span programs*. The class PPA [46] is a well-known subclass of TFNP embodying the combinatorial principle “every graph with an odd degree vertex has another”. Informally, a search problem is in PPA if for every  $n$ -bit input  $x$  we may describe implicitly an undirected graph  $G_x = (V, E)$  (typically of size exponential in  $n$ ; the edge relation is computed by a polynomial-size circuit) such that  $G$  has degree at most 2, there is a distinguished degree-1 vertex  $v^* \in V$ , and every other degree-1 vertex  $v \in V$  is associated with a feasible solution to the instance  $x$  (that is, the solution can be efficiently computed from  $v$ ).



**Communication PPA.** The communication analogue of PPA is defined canonically by letting the edge relation be computed by a (deterministic) communication protocol. Specifically, first fix a two-party search problem  $S \subseteq \mathcal{X} \times \mathcal{Y} \times \mathcal{O}$ , that is, Alice gets  $x \in \mathcal{X}$ , Bob gets  $y \in \mathcal{Y}$ , and their goal is to find a *feasible solution* in  $S(x, y) := \{o \in \mathcal{O} : (x, y, o) \in S\}$ . A PPA-protocol  $\Pi$  solving  $S$  consists of a vertex set  $V$ , a distinguished vertex  $v^* \in V$ , and for each vertex  $v \in V$  there is an associated solution  $o_v \in \mathcal{O}$  and a protocol  $\Pi_v$  (taking inputs from  $\mathcal{X} \times \mathcal{Y}$ ). Given an input  $(x, y)$ , the protocols  $\Pi_v$  implicitly describe a graph  $G = G_{x,y}$

on the vertex set  $V$  as follows. The output of protocol  $\Pi_v$  on input  $(x, y)$  is interpreted as a subset  $\Pi_v(x, y) \subseteq V$  of size at most 2. We define  $\{u, v\} \in E(G)$  iff  $u \in \Pi_v(x, y)$  and  $v \in \Pi_u(x, y)$ . The correctness requirements are:

(C1) if  $\deg(v^*) \neq 1$ , then  $o_{v^*} \in S(x, y)$ .

(C2) if  $\deg(v) \neq 2$  for  $v \neq v^*$ , then  $o_v \in S(x, y)$ .

The *cost* of  $\Pi$  is defined as  $\log |V| + \max_v |\Pi_v|$  where  $|\Pi_v|$  is the communication cost of  $\Pi_v$ . Finally, define  $\text{PPA}^{\text{cc}}(S)$  as the least cost of a PPA-protocol that solves  $S$ .

For a (monotone) function  $f$ , define  $\text{SP}_{\mathbb{F}}(f)$  (resp.  $\text{mSP}_{\mathbb{F}}(f)$ ) as the least size of a (monotone)  $\mathbb{F}$ -span program computing  $f$ . Our characterization is in terms of  $S := \text{KW}(f)$ .

► **Theorem 3.** *For any boolean function  $f$ , we have  $\log \text{SP}_{\mathbb{F}_2}(f) = \Theta(\text{PPA}^{\text{cc}}(\text{KW}(f)))$ . Furthermore, if  $f$  is monotone, we have  $\log \text{mSP}_{\mathbb{F}_2}(f) = \Theta(\text{PPA}^{\text{cc}}(\text{KW}^+(f)))$ .*

**Query PPA.** Our second characterization concerns the *Nullstellensatz* proof system; see Section 3 for the standard definition. Span programs and Nullstellensatz are known to be connected via interpolation [51] and lifting [48]. Given our first characterization (Theorem 3), it is no surprise that a companion result should hold in query complexity: *the query complexity analogue of PPA captures the degree of Nullstellensatz refutations over  $\mathbb{F}_2$ .*

The query analogue of PPA is defined in the same way as the communication analogue, except we replace protocols by (deterministic) decision trees. In fact, query PPA was already studied by Beame et al. [6] who separated query analogues of different subclasses of TFNP. To define it, first fix a search problem  $S \subseteq \{0, 1\}^n \times \mathcal{O}$ , that is, on input  $x \in \{0, 1\}^n$  the goal is to find a *feasible solution* in  $S(x) := \{o \in \mathcal{O} : (x, o) \in S\}$ . A *PPA-decision tree*  $\mathcal{T}$  solving  $S$  consists of a vertex set  $V$ , a distinguished vertex  $v^* \in V$ , and for each vertex  $v \in V$  there is an associated solution  $o_v \in \mathcal{O}$  and a decision tree  $\mathcal{T}_v$  (querying bits of an  $n$ -bit input). Given an input  $x \in \{0, 1\}^n$ , the decision trees  $\mathcal{T}_v$  implicitly describe a graph  $G = G_x$  on the vertex set  $V$  as follows. The output of  $\mathcal{T}_v$  on input  $x$  is interpreted as a subset  $\mathcal{T}_v(x) \subseteq V$  of size at most 2. We then define  $\{u, v\} \in E(G)$  iff  $u \in \mathcal{T}_v(x)$  and  $v \in \mathcal{T}_u(x)$ . The correctness requirements are the same as before, 1 and 2. The *cost* of  $\mathcal{T}$  is defined as the maximum over all  $v \in V$  and all inputs  $x$  of the number of queries made by  $\mathcal{T}_v$  on input  $x$ . Finally, define  $\text{PPA}^{\text{dt}}(S)$  as the least cost of a PPA-decision tree that solves  $S$ .

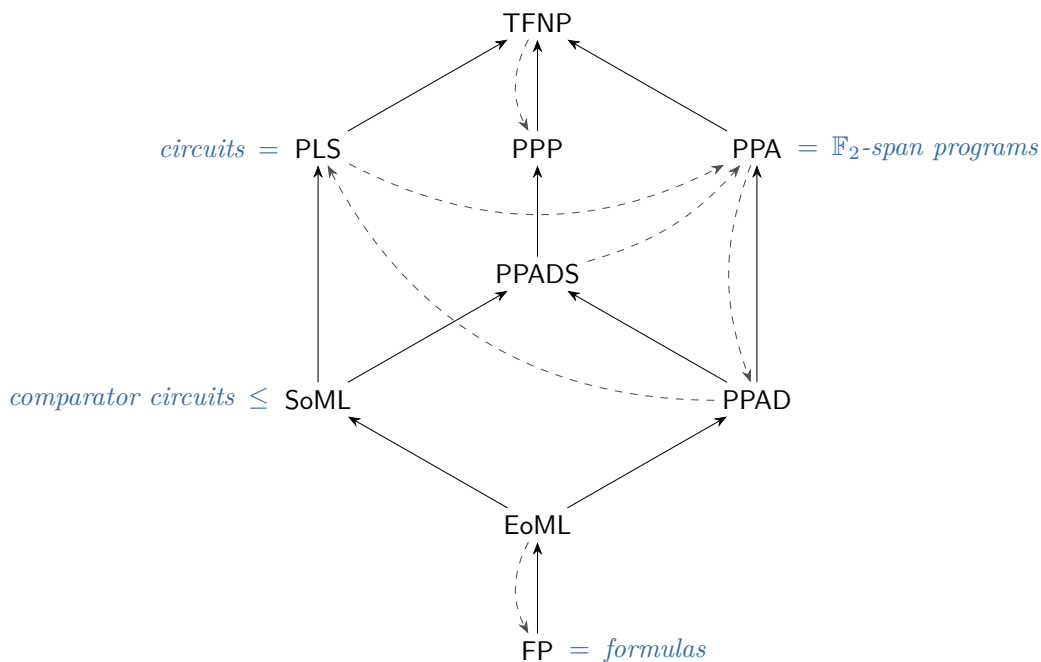
With any unsatisfiable  $n$ -variate boolean CSP  $F$  one can associate a canonical search problem:

**CSP search problem  $S(F)$**  = *input*: an  $n$ -variate truth assignment  $x \in \{0, 1\}^n$   
*output*: constraint  $C$  of  $F$  falsified by  $x$  (i.e.,  $C(x) = 0$ )

► **Theorem 4.** *The  $\mathbb{F}_2$ -Nullstellensatz degree of an  $k$ -CNF formula  $F$  equals  $\Theta(\text{PPA}^{\text{dt}}(S(F)))$ .*

The easy direction of this characterization is that Nullstellensatz degree lower bounds  $\text{PPA}^{\text{dt}}$ . This fact was already observed and exploited by Beame et al. [6] to prove lower bounds for  $\text{PPA}^{\text{dt}}$ . Our contribution is to show the other (less trivial) direction.

Let us finally mention a related result in Turing machine complexity due to Belovs et al. [9]: a circuit-encoded version of Nullstellensatz is PPA-complete. Their proof is highly nontrivial whereas our characterizations admit relatively short proofs, owing partly to us working with simple nonuniform models of computation.



■ **Figure 1** The landscape of communication search problem classes (uncluttered by the usual ‘cc’ superscripts). A solid arrow  $C_1 \rightarrow C_2$  denotes  $C_1 \subseteq C_2$ , and a dashed arrow  $C_1 \dashrightarrow C_2$  denotes  $C_1 \not\subseteq C_2$  (in fact, an exponential separation). Some classes can characterize other models of computation (printed in blue). See Appendix A for definitions.

## 2 Survey: Communication TFNP

Given the results in Section 1, it is natural to examine other communication analogues of subclasses of TFNP. The goal in this section is to explain the current state of knowledge as summarized in Figure 1. The formal definitions of the communication classes appear in Appendix A.

**TFNP.** As is customary in structural communication complexity [2, 30, 27] we formally define  $\text{TFNP}^{\text{cc}}$  (resp.  $\text{PLS}^{\text{cc}}$ ,  $\text{PPA}^{\text{cc}}$ , etc.) as the class of all total two-party  $n$ -bit search problems that admit a nondeterministic protocol<sup>5</sup> (resp. PLS-protocol, PPA-protocol, etc.) of communication cost  $\text{polylog}(n)$ . For example, Karchmer–Wigderson games  $\text{KW}(f)$  and  $\text{KW}^+(f)$ , for an  $n$ -bit boolean function  $f$ , have efficient nondeterministic protocols: guess a  $\log n$ -bit coordinate  $i \in [n]$  and check that  $x_i \neq y_i$  or  $x_i > y_i$ . Hence these problems are in  $\text{TFNP}^{\text{cc}}$ . In fact, a converse holds: any total two-party search problem with nondeterministic complexity  $c$  can be reduced to  $\text{KW}^+(f)$  for some  $2^c$ -bit *partial* monotone function  $f$ , see [21, Lemma 2.3]. In summary, the study of total NP search problems in communication complexity is equivalent to the study of monotone Karchmer–Wigderson games for *partial* monotone functions.

<sup>5</sup> That is, for any input  $(x, y)$ , every accepting computation of the nondeterministic protocol outputs a feasible solution  $o \in \mathcal{O}$  for  $(x, y)$ . An alternative, more restrictive definition of  $\text{TFNP}^{\text{cc}}$  (which is closer to how the classical class is defined) is to require that there is an efficient deterministic protocol that on input  $(x, y)$  decides whether  $o \in \mathcal{O}$  is feasible for  $(x, y)$ . In this paper we stick with the stronger (and simpler) definition for convenience. All results hold equally well under the more restrictive definition.

Sometimes a *partial* function  $f$  can be canonically extended into a *total* one  $f'$  without increasing the complexity of  $\text{KW}(f)$  (or  $\text{KW}^+(f)$ ). This is possible whenever  $\text{KW}(f)$  lies in a communication class that captures some associated model of computation. For example, if  $\text{KW}(f)$  is solved by a deterministic protocol (resp. PLS-protocol, PPA-protocol) then the Karchmer–Wigderson connection can build us a corresponding formula (resp. circuit,  $\mathbb{F}_2$ -span program) that computes some total extension  $f'$  of  $f$ . Consequently, separating two communication classes that capture two monotone models is *equivalent* to separating the monotone models themselves.

**FP.** Raz and McKenzie [52] showed an exponential separation between monotone formula size and monotone circuit size. This can be rephrased as  $\text{PLS}^{\text{cc}} \not\subseteq \text{FP}^{\text{cc}}$ . Their technique is much more general: they develop a query-to-communication lifting theorem for deterministic protocols (see also [26] for exposition). By plugging in known query complexity lower bounds against the class  $\text{EoML}$  (combinatorial subclass of  $\text{CLS}$  [17] introduced by [32, 19]), one can obtain a stronger separation  $\text{EoML}^{\text{cc}} \not\subseteq \text{FP}^{\text{cc}}$ .

A related question is whether *randomization* helps in solving  $\text{TFNP}^{\text{cc}}$  problems. Lower bounds against randomized protocols have applications in proof complexity [34, 7, 33, 25] and algorithmic game theory [58, 5, 28, 57, 22, 4]. In particular, some of these works (for finding Nash equilibria) have introduced a communication analogue of the PPAD-complete END-OF-LINE problem, which we will continue to study in Subsection 4.2.

**PLS.** Razborov’s [54] famous monotone circuit lower bound for the *clique/coloring* problem (which is in  $\text{PPP}^{\text{cc}}$ ) can be interpreted as an exponential separation  $\text{PPP}^{\text{cc}} \not\subseteq \text{PLS}^{\text{cc}}$ . We show a stronger separation  $\text{PPAD}^{\text{cc}} \not\subseteq \text{PLS}^{\text{cc}}$  using the END-OF-LINE problem in Subsection 4.2. Note that this is even slightly stronger than Theorem 1, which only implies  $\text{PPA}^{\text{cc}} \not\subseteq \text{PLS}^{\text{cc}}$ .

**PPApD.** In light of our characterization of  $\text{PPA}^{\text{cc}}$ , we may interpret the inability of monotone  $\mathbb{F}_2$ -span program to efficiently simulate monotone circuits [48] as a separation  $\text{PLS}^{\text{cc}} \not\subseteq \text{PPA}^{\text{cc}}$ . We show an incomparable separation  $\text{PPADS}^{\text{cc}} \not\subseteq \text{PPA}^{\text{cc}}$  in Subsection 4.3.

In the other direction, prior work implies  $\text{PPA}^{\text{cc}} \not\subseteq \text{PPAD}^{\text{cc}}$  as follows. Pitassi and Robere [48] exhibit a monotone  $f$  (in hindsight, one can take  $f := 3\text{XOR-SAT}_n$ ) computable with a small monotone  $\mathbb{F}_2$ -span program (hence  $\text{KW}^+(f) \in \text{PPA}^{\text{cc}}$ ) and such that  $\text{KW}^+(f)$  has an exponentially large  $\mathbb{R}$ -partition number (see Section 3 for a definition); however, we observe that all problems in  $\text{PPAD}^{\text{cc}}$  have a small  $\mathbb{R}$ -partition number (see Remark 9).

**PPP.** There are no lower bounds against  $\text{PPP}^{\text{cc}}$  for an *explicit* problem in  $\text{TFNP}^{\text{cc}}$ . However, we can show non-constructively the existence of  $\text{KW}(f) \in \text{TFNP}^{\text{cc}}$  such that  $\text{KW}(f) \notin \text{PPP}^{\text{cc}}$ , which implies  $\text{PPP}^{\text{cc}} \neq \text{TFNP}^{\text{cc}}$ . Indeed, we argue in Remark 8 that every  $S$  reduces to  $\text{KW}^+(3\text{CNF-SAT}_N)$  over  $N := \exp(O(\text{PPP}^{\text{cc}}(S)))$  variables. Applying this to  $S := \text{KW}(f)$  for an  $n$ -bit  $f$ , we conclude that  $f$  is a (non-monotone) projection of  $3\text{CNF-SAT}_N$  for  $N := \exp(O(\text{PPP}^{\text{cc}}(\text{KW}(f))))$ . In particular, if  $\text{KW}(f) \in \text{PPP}^{\text{cc}}$  (i.e.,  $\text{PPP}^{\text{cc}}(\text{KW}(f)) \leq \text{polylog}(n)$ ), then  $f$  is in non-uniform quasipoly-size NP. Therefore  $\text{KW}(f) \notin \text{PPP}^{\text{cc}}$  for a random  $f$ .

**EoML, SoML, and comparator circuits.** One prominent circuit model that currently lacks a characterization via a  $\text{TFNP}^{\text{cc}}$  subclass is *comparator circuits* [41, 16]. These circuits are composed only of *comparator gates* (taking two input bits and outputting them in sorted order) and input literals (positive literals in the monotone case).

We can show an upper bound better than  $\text{PLS}^{\text{cc}}$  for comparator circuits. Indeed, we introduce a new class  $\text{SoML}$  generalizing  $\text{EoML}$  [32, 19] as follows. Recall that  $\text{EoML}$  is the class of problems reducible to  $\text{END-OF-METERED-LINE}$ : we are given a directed graph of in/out-degree at most 1 with a distinguished source vertex  $v^*$  (in-degree 0), and moreover, each vertex is labeled with an integer “meter” that is strictly decreasing along directed paths; a solution is any sink or source distinct from  $v^*$ . The complete problem defining  $\text{SoML}$  is  $\text{SINK-OF-METERED-LINE}$ , which is the same as  $\text{END-OF-METERED-LINE}$  except only sinks count as solutions. It is not hard (left as an exercise) to adapt the characterization of circuits via  $\text{PLS}^{\text{cc}}$  [55, 50, 60] to show that  $\text{KW}(f)$  is in  $\text{SoML}^{\text{cc}}$  if  $f$  is computed by a small comparator circuit. However, we suspect that the converse ( $\text{SoML}$ -protocol for  $\text{KW}(f)$  implies a comparator circuit) is false.

## 2.1 Open problems

In query complexity, the relative complexities of TFNP subclasses are almost completely understood [6, 12, 43]. In communication complexity, by contrast, there are huge gaps in our understanding as can be gleaned from Figure 1. For example:

- (1) There are no lower bounds against classes  $\text{PPADS}^{\text{cc}}$  and  $\text{PPP}^{\text{cc}}$  for an explicit problem in  $\text{TFNP}^{\text{cc}}$ . For starters, show  $\text{PLS}^{\text{cc}} \not\subseteq \text{PPADS}^{\text{cc}}$  or  $\text{PPA}^{\text{cc}} \not\subseteq \text{PPADS}^{\text{cc}}$ .
- (2) Find computational models captured by  $\text{EoML}^{\text{cc}}$ ,  $\text{SoML}^{\text{cc}}$ ,  $\text{PPAD}^{\text{cc}}$ ,  $\text{PPADS}^{\text{cc}}$ ,  $\text{PPP}^{\text{cc}}$ .
- (3) Query-to-communication lifting theorems are known for  $\text{FP}$  [52],  $\text{PLS}$  [23],  $\text{PPA}$  [48]. Prove more. (This is one way to attack Question 1 if proved for  $\text{PPADS}$ .)
- (4) Prove more separations. For example, can our result  $\text{PPADS}^{\text{cc}} \not\subseteq \text{PPA}^{\text{cc}}$  be strengthened to  $\text{SoML}^{\text{cc}} \not\subseteq \text{PPA}^{\text{cc}}$ ? This is closely related to whether monotone comparator circuits can be more powerful than monotone  $\mathbb{F}_2$ -span programs (no separation is currently known).

## 3 Preliminaries

**C-Sat.** Fix an alphabet  $\Sigma$  (potentially infinite, e.g.,  $\Sigma = \mathbb{R}$ ). Let  $\mathcal{C}$  be a finite set of  $k$ -ary predicates over  $\Sigma$ , that is, each  $C \in \mathcal{C}$  is a function  $C: \Sigma^k \rightarrow \{0, 1\}$ . We define a monotone function  $\mathcal{C}\text{-SAT}_n: \{0, 1\}^N \rightarrow \{0, 1\}$  over  $N = |\mathcal{C}|n^k$  input bits as follows. An input  $x \in \{0, 1\}^N$  is interpreted as a  $\mathcal{C}$ -CSP instance, that is,  $x$  is (the indicator vector of) a set of  $\mathcal{C}$ -constraints, each applied to a  $k$ -tuple of variables from  $v_1, \dots, v_n$ . We define  $\mathcal{C}\text{-SAT}_n(x) := 1$  iff the  $\mathcal{C}$ -CSP  $x$  is *unsatisfiable*: no assignment  $v \in \Sigma^n$  exists such that  $C(v) = 1$  for all  $C \in x$ .

For a field  $\mathbb{F}$ , we define  $k\text{LIN}(\mathbb{F})$  as the set of all  $\mathbb{F}$ -linear equations of the form

$$\sum_{i \in [k]} a_i v_i = a_0, \quad \text{where } a_i \in \{0, \pm 1\}.$$

In particular, we recover  $3\text{XOR-SAT}_n$  defined in Section 1 essentially as  $3\text{LIN}(\mathbb{F}_2)\text{-SAT}_n$ . We could have allowed the  $a_i$  to range over  $\mathbb{F}$  when  $\mathbb{F}$  is finite, but we stick with the above convention as it ensures that the set  $k\text{LIN}(\mathbb{R})$  is always finite.

**Boolean alphabets.** We assume henceforth that all alphabets  $\Sigma$  contain distinguished elements 0 and 1. We define  $\mathcal{C}_{\text{bool}}$  to be the constraint set obtained from  $\mathcal{C}$  by restricting each  $C \in \mathcal{C}$  to the boolean domain  $\{0, 1\}^k \subseteq \Sigma^k$ . Moreover, if  $F$  is a  $\mathcal{C}$ -CSP, we write  $F_{\text{bool}}$  for the  $\mathcal{C}_{\text{bool}}$ -CSP obtained by restricting the constraints of  $F$  to boolean domains. Consequently, any  $S(F_{\text{bool}})$  associated with a  $\mathcal{C}$ -CSP  $F$  is a *boolean* search problem.

**Algebraic partitions.** We say that a subset  $A \subseteq \mathcal{X} \times \mathcal{Y}$  is *monochromatic* for a two-party search problem  $S \subseteq \mathcal{X} \times \mathcal{Y} \times \mathcal{O}$  if there is some  $o \in \mathcal{O}$  such that  $o \in S(x, y)$  for all  $(x, y) \in A$ . Moreover, if  $M \in \mathbb{F}^{\mathcal{X} \times \mathcal{Y}}$  is a matrix, we say  $M$  is *monochromatic* if the support of  $M$  is monochromatic. For any field  $\mathbb{F}$ , an  $\mathbb{F}$ -*partition* of a search problem  $S$  is a set  $\mathcal{M}$  of rank-1 matrices  $M \in \mathbb{F}^{\mathcal{X} \times \mathcal{Y}}$  such that  $\sum_{M \in \mathcal{M}} M = \mathbb{1}$  and each  $M \in \mathcal{M}$  is monochromatic for  $S$ . The *size* of the partition is  $|\mathcal{M}|$ . The  $\mathbb{F}$ -*partition number*  $\chi_{\mathbb{F}}(S)$  is the least size of an  $\mathbb{F}$ -partition of  $S$ . In the following characterization, recall that we use  $\text{SP}_{\mathbb{F}}$  and  $\text{mSP}_{\mathbb{F}}$  to denote (monotone) span program complexity.

► **Theorem 5** ([21]). *For any boolean function  $f$  and any field  $\mathbb{F}$ ,  $\text{SP}_{\mathbb{F}}(f) = \chi_{\mathbb{F}}(\text{KW}(f))$ . Furthermore, if  $f$  is monotone then  $\text{mSP}_{\mathbb{F}}(f) = \chi_{\mathbb{F}}(\text{KW}^+(f))$ .*

**Nullstellensatz.** Let  $P := \{p_1 = 0, p_2 = 0, \dots, p_m = 0\}$  be an unsatisfiable system of polynomial equations in  $\mathbb{F}[z_1, z_2, \dots, z_n]$  for a field  $\mathbb{F}$ . An  $\mathbb{F}$ -*Nullstellensatz refutation* of  $P$  is a sequence of polynomials  $q_1, q_2, \dots, q_m \in \mathbb{F}[z_1, z_2, \dots, z_n]$  such that  $\sum_{i=1}^m q_i p_i = 1$  where the equality is syntactic. The *degree* of the refutation is  $\max_i \deg(q_i p_i)$ . The  $\mathbb{F}$ -*Nullstellensatz degree* of  $P$ , denoted  $\text{NS}_{\mathbb{F}}(P)$ , is the least degree of an  $\mathbb{F}$ -Nullstellensatz refutation of  $P$ .

Moreover, if  $F$  is a  $k$ -CNF formula (or a boolean  $k$ -CSP), we often tacitly think of it as a polynomial system  $P_F$  by using the standard encoding (e.g.,  $(z_1 \vee \neg z_2) \rightsquigarrow (1 - z_1)z_2 = 0$ ) and also including the *boolean axioms*  $z_i^2 - z_i = 0$  in  $P_F$  if we are working over  $\mathbb{F} \neq \mathbb{F}_2$ .

**Lifting theorems.** Let  $S \subseteq \{0, 1\}^n \times \mathcal{O}$  be a boolean search problem and  $g: \mathcal{X} \times \mathcal{Y} \rightarrow \{0, 1\}$  a two-party function, usually called a *gadget*. The composed search problem  $S \circ g^n \subseteq \mathcal{X}^n \times \mathcal{Y}^n \times \mathcal{O}$  is defined as follows: Alice holds  $x \in \mathcal{X}^n$ , Bob holds  $y \in \mathcal{Y}^n$ , and their goal is to find an  $o \in S(z)$  where  $z := g^n(x, y) = (g(x_1, y_1), \dots, g(x_n, y_n))$ . We focus on the usual *index gadget*  $\text{IND}_m: [m] \times \{0, 1\}^m \rightarrow \{0, 1\}$  given by  $\text{IND}_m(x, y) := y_x$ .

The main results of [23, 48] can be summarized as follows (we define more terms below).

► **Theorem 6.** *Let  $k \geq 1$  be a constant and let  $m = m(n) := n^C$  for a large enough constant  $C \geq 1$ . Then for any an unsatisfiable boolean  $n$ -variate  $k$ -CSP  $F$ ,*

$$\begin{aligned} [23]: \quad & \text{PLS}^{\text{cc}}(S(F) \circ \text{IND}_m^n) = \text{PLS}^{\text{dt}}(S(F)) \cdot \Theta(\log n), \\ [48]: \quad & \text{PPA}^{\text{cc}}(S(F) \circ \text{IND}_m^n) = \text{PPA}^{\text{dt}}(S(F)) \cdot \Theta(\log n), \\ [48]: \quad & \log \chi_{\mathbb{F}}(S(F) \circ \text{IND}_m^n) = \text{NS}_{\mathbb{F}}(F) \cdot \Theta(\log n), \quad \forall \mathbb{F} \in \{\mathbb{F}_p, \mathbb{R}\}. \end{aligned}$$

For aesthetic reasons, we have used  $\text{PLS}^{\text{dt}}(S(F))$  here to denote the *Resolution width* of  $F$  (introduced in [10]), which is how the result of [23] was originally stated. (But one can check that the query analogue of PLS, obtained by replacing protocols with decision trees in Definition 14, is indeed equivalent to Resolution width.) We also could not resist incorporating our new characterizations of  $\text{PPA}^{\text{cc}}$  and  $\text{PPA}^{\text{dt}}$  to interpret the result of [48] specialized to  $\mathbb{F}_2$ .

## 4 Proofs of Separations

In this section, we show lower bounds for  $\mathcal{C}$ -SAT against monotone circuits (Theorem 1) and monotone span programs (Theorem 2), plus some bonus results ( $\text{PPAD}^{\text{cc}} \not\subseteq \text{PLS}^{\text{cc}}$ ,  $\text{PPADS}^{\text{cc}} \not\subseteq \text{PPA}^{\text{cc}}$ , Nullstellensatz degree vs. Cutting Planes).

## 4.1 Reduction

The key to our lower bounds is a new reduction. We show that a lifted version of  $S(F_{\text{bool}})$ , where  $F$  is an unsatisfiable  $\mathcal{C}$ -CSP, reduces to the monotone Karchmer–Wigderson game for  $\mathcal{C}$ -SAT. Note that we require  $F$  to be unsatisfiable over its original alphabet  $\Sigma$ , but the reduction is from the booleanized (and hence easier-to-refute) version of  $F$ .

► **Lemma 7.** *Let  $F$  be an unsatisfiable  $\mathcal{C}$ -CSP. Then  $S(F_{\text{bool}}) \circ \text{IND}_m^n$  reduces to  $\text{KW}^+(\mathcal{C}\text{-SAT}_{nm})$ .*

**Proof.** Suppose the  $\mathcal{C}$ -CSP  $F$  consists of  $k$ -ary constraints  $C_1, \dots, C_t$  applied to variables  $z_1, \dots, z_n$ . We reduce  $S(F_{\text{bool}}) \circ \text{IND}_m^n \subseteq [m]^n \times (\{0, 1\}^m)^n \times [t]$  to the problem  $\text{KW}^+(f) \subseteq f^{-1}(1) \times f^{-1}(0) \times [N]$  where  $f := \mathcal{C}\text{-SAT}_{mn}$  over  $N := |\mathcal{C}|(mn)^k$  input bits. The two parties compute locally as follows.

**Alice:** Given  $(x_1, \dots, x_n) \in [m]^n$ , Alice constructs a  $\mathcal{C}$ -CSP over variables  $\{v_{i,j} : (i, j) \in [n] \times [m]\}$  that is obtained from  $F$  by renaming its variables  $z_1, \dots, z_n$  to  $v_{1,x_1}, \dots, v_{n,x_n}$  (in this order). Since  $F$  was unsatisfiable, so is Alice’s variable-renamed version of it. Thus, when interpreted as an indicator vector of constraints, Alice has constructed a 1-input of  $\mathcal{C}\text{-SAT}_{mn}$ .

**Bob:** Given  $y \in (\{0, 1\}^m)^n$ , Bob constructs a  $\mathcal{C}$ -CSP over variables  $\{v_{i,j} : (i, j) \in [n] \times [m]\}$  as follows. We view  $y$  naturally as a boolean assignment to the variables  $v_{i,j}$ . Bob includes in his  $\mathcal{C}$ -CSP instance all possible  $\mathcal{C}$ -constraints  $C$  applied to the  $v_{i,j}$  such that  $C$  is satisfied under the assignment  $y$  (i.e.,  $C(y) = 1$ ). This is clearly a satisfiable  $\mathcal{C}$ -CSP instance, as the assignment  $y$  satisfies all Bob’s constraints. Thus, when interpreted as an indicator vector of constraints, Bob has constructed a 0-input of  $\mathcal{C}\text{-SAT}_{mn}$ .

It remains to argue that any solution to  $\text{KW}^+(\mathcal{C}\text{-SAT}_{mn})$  gives rise to a solution to  $S(F_{\text{bool}}) \circ \text{IND}_m^n$ . Indeed, a solution to  $\text{KW}^+(\mathcal{C}\text{-SAT}_{mn})$  corresponds to a  $\mathcal{C}$ -constraint  $C$  that is present in Alice’s  $\mathcal{C}$ -CSP but not in Bob’s. By Bob’s construction, such a  $C$  must be violated by the assignment  $y$  (i.e.,  $C(y) = 0$ ). Since all Alice’s constraints involve only variables  $v_{1,x_1}, \dots, v_{n,x_n}$ , the constraint  $C$  must in fact be violated by the partial assignment to the said variables, which is  $z = \text{IND}_m^n(x, y)$ . Thus the constraint of  $F$  from which  $C$  was obtained via renaming is a solution to  $S(F_{\text{bool}}) \circ \text{IND}_m^n$ . ◀

► **Remark 8 (Generic reduction to CNF-SAT).** We claim that any problem  $S \subseteq \mathcal{X} \times \mathcal{Y} \times \mathcal{O}$  that lies in one of the known subclasses of  $\text{TFNP}^{\text{cc}}$  (as listed in Section 2) reduces efficiently to  $\text{KW}^+(k\text{CNF-SAT}_n)$  for constant  $k$  (one can even take  $k = 3$  by standard reductions). Let us sketch the argument for  $S \in \text{PPP}^{\text{cc}}$ ; after all, better reductions are known for  $\text{PLS}^{\text{cc}}$  and  $\text{PPA}^{\text{cc}}$ , namely to  $\text{HORN-SAT}$  and  $3\text{XOR-SAT}$ .

**Proof Sketch.** Let  $\Pi := (V, v^*, o_v, \Pi_v)$  be a PPP-protocol solving  $S$  of cost  $c := \text{PPP}^{\text{cc}}(S)$ . We may assume wlog that all the  $\Pi_v$  have constant communication cost  $k \leq O(1)$  by embedding the protocol trees of the  $\Pi_v$  as part of the implicitly described bipartite graph. In particular, we view each  $\Pi_v$  as a function  $\mathcal{X} \times \mathcal{Y} \rightarrow \{0, 1\}^k$  where the output is interpreted according to some fixed map  $\{0, 1\}^k \rightarrow V$ . Consider a set of  $n := k|V|$  ( $|V| \leq 2^c$ ) boolean variables  $\{z_{v,i} : (v, i) \in V \times [k]\}$  with the intuitive interpretation that  $z_{v,i}$  is the  $i$ -th output bit of  $\Pi_v$ . We may encode the correctness conditions for  $\Pi$  as an unsatisfiable  $2k$ -CNF formula  $F$  over the  $z_{v,i}$  that has, for each  $\{v, u\} \in \binom{V}{2}$ , clauses requiring that the outputs of  $\Pi_v$  and  $\Pi_u$  (as encoded by the  $z_{v,i}$ ) should point to distinct vertices. Finally, we note that computing the  $i$ -th output bit  $(\Pi_v)_i : \mathcal{X} \times \mathcal{Y} \rightarrow \{0, 1\}$  reduces to a large enough constant-size index gadget  $\text{IND}_{O(1)}$  (which embeds any two-party function of communication complexity  $k \leq O(1)$ ). Therefore  $S$  naturally reduces to  $S(F) \circ \text{IND}_{O(1)}^n$ , which by Lemma 7 reduces to  $\text{KW}^+(2k\text{CNF-SAT}_{O(n)})$ , as desired. ◀



## 4.2 Monotone circuit lower bounds

**Xor-Sat.** The easiest result to prove is Theorem 1: an exponential monotone circuit lower bound for 3XOR-SAT<sub>n</sub>. By the characterization of [55] it suffices to show

$$\text{PLS}^{\text{cc}}(\text{KW}^+(\text{3XOR-SAT}_n)) \geq n^{\Omega(1)}. \quad (1)$$

Urquhart [62] exhibited unsatisfiable  $n$ -variate 3XOR-CSPs  $F$  (aka *Tseitin formulas*) requiring linear Resolution width, that is,  $\text{PLS}^{\text{dt}}(S(F)) \geq \Omega(n)$  in our notation. Hence Theorem 6 implies that  $\text{PLS}^{\text{cc}}(S(F) \circ \text{IND}_m^n) \geq \Omega(n)$  for some  $m = n^{O(1)}$ . By the reduction in Lemma 7, we get that  $\text{PLS}^{\text{cc}}(\text{KW}^+(\text{3XOR-SAT}_{nm})) \geq \Omega(n)$ . (Note that 3XOR has a boolean alphabet, so  $F = F_{\text{bool}}$ .) This yields the claim (1) by reparameterizing the number of variables.

**Lin(F)-Sat.** More generally, we can prove a similar lower bound over any field  $\mathbb{F} \in \{\mathbb{F}_p, \mathbb{R}\}$ :

$$\text{PLS}^{\text{cc}}(\text{KW}^+(\text{3LIN}(\mathbb{F})\text{-SAT}_n)) \geq n^{\Omega(1)}. \quad (2)$$

Fix such an  $\mathbb{F}$  henceforth. This time we start with a  $k\text{LIN}(\mathbb{F})$ -CSP introduced in [13] for  $\mathbb{F} = \mathbb{F}_p$  (aka *mod- $p$  Tseitin formulas*), but the definition generalizes to any field. The CSP is constructed based on a given directed graph  $G = (V, E)$  that is *regular*:  $\text{in-deg}(v) = \text{out-deg}(v) = k/2$  for all  $v \in V$ . Fix also a distinguished vertex  $v^* \in V$ . Then  $F = F_{G, \mathbb{F}}$  is defined as the following  $k\text{LIN}(\mathbb{F})$ -CSP over variables  $\{z_e : e \in E\}$ :

$$\forall v \in V : \sum_{(v,u) \in E} z_{(v,u)} - \sum_{(u,v) \in E} z_{(u,v)} = \mathbf{1}_{v^*}(v), \quad (F_{G, \mathbb{F}})$$

where  $\mathbf{1}_{v^*}(v^*) = 1$  and  $\mathbf{1}_{v^*}(v) = 0$  for  $v \neq v^*$ . This system is unsatisfiable because the sum over  $v \in V$  of the RHS equals 1 whereas the sum of the LHS equals 0 (each variable appears once with a positive sign, once with a negative sign).

We claim that the booleanized  $k$ -CSP  $F_{\text{bool}}$  (more precisely, its natural  $k$ -CNF encoding) has linear Resolution width, that is,  $\text{PLS}^{\text{dt}}(S(F_{\text{bool}})) \geq \Omega(n)$  in our notation. Indeed, the constraints of  $F_{\text{bool}}$  are  $k/2$ -robust in the sense that if a partial assignment  $\rho \in \{0, 1, *\}^k$  fixes the value of a constraint of  $F_{\text{bool}}$ , then  $\rho$  must set more than  $k/2$  variables. Alekhovich et al. [1, Theorem 3.1] show that if  $k$  is a large enough constant, there exist regular expander graphs  $G$  such that  $F_{\text{bool}}$  (or any  $k$ -CSP with  $\Omega(k)$ -robust constraints) has Resolution width  $\Omega(n)$ , as desired.

Combining the above with the lifting theorem in Theorem 6 and the reduction in Lemma 7 yields  $\text{PLS}^{\text{cc}}(k\text{LIN}(\mathbb{F})\text{-SAT}_n) \geq n^{\Omega(1)}$  for large enough  $k$ . Finally, we can reduce the arity from  $k$  to 3 by a standard trick. For example, given the linear constraint  $a_1v_1 + a_2v_2 + a_3v_3 + a_4v_4 = a_0$  we can introduce a new auxiliary variable  $u$  and two equations  $a_1v_1 + a_2v_2 + u = 0$  and  $-u + a_3v_3 + a_4v_4 = a_0$ . In general, we replace each equation on  $k > 3$  variables with a collection of  $k - 2$  equations by introducing  $k - 3$  auxiliary variables to create an equisatisfiable instance. This shows that  $k\text{LIN}(\mathbb{F})\text{-SAT}_n$  reduces to (i.e., is a monotone projection of)  $3\text{LIN}(\mathbb{F})\text{-SAT}_{kn}$ , which concludes the proof of (2).

**PPAD<sup>cc</sup>  $\not\leq$  PLS<sup>cc</sup> via End-of-Line.** Consider the  $\mathbb{R}$ -linear system  $F = F_{G, \mathbb{R}}$  defined above. We observe that  $S(F_{\text{bool}})$  is in fact equivalent to (a query version of) the PPAD-complete END-OF-LINE problem. In the END-OF-LINE problem, we are given a directed graph of in/out-degree at most 1 and a distinguished source vertex  $v^*$  (in-degree 0); the goal is to find a sink or a source distinct from  $v^*$  (cf. Definition 15). On the other hand, in  $S(F_{\text{bool}})$

we are given a boolean assignment  $z \in \{0, 1\}^E$ , which can be interpreted as (the indicator vector of) a subset of edges defining a (spanning) subgraph  $G_z$  of  $G$ ; the goal is to find a vertex  $v \in V$  such that either

- (1)  $v = v^*$  and  $\text{out-deg}(v) \neq \text{in-deg}(v) + 1$  in  $G_z$ ; or
- (2)  $v \neq v^*$  and  $\text{out-deg}(v) \neq \text{in-deg}(v)$  in  $G_z$ .

The only essential difference between  $S(F_{\text{bool}})$  and END-OF-LINE is that the graph  $G_z$  can have in/out-degree a large constant  $k/2$  rather than 1. But there is a standard reduction between the two problems [46]: we may locally interpret a vertex  $v \in V(G_z)$  with  $\text{out-deg}(v) = \text{in-deg}(v) = \ell$  as  $\ell$  distinct vertices of in/out-degree 1. This reduction also shows that the lifted problem  $S(F_{\text{bool}}) \circ \text{IND}_m$  for  $m = n^{O(1)}$  admits a  $O(\log n)$ -cost PPAD-protocol, and is thus in  $\text{PPAD}^{\text{cc}}$ . By contrast, we proved above that this problem is not in  $\text{PLS}^{\text{cc}}$  (for appropriate  $G$ ).

► **Remark 9 (Algebraic partitions for  $\text{PPAD}^{\text{cc}}$ ).** We claim that every problem  $S \in \text{PPAD}^{\text{cc}}$  admits a small  $\mathbb{Z}$ -partition, and hence a small  $\mathbb{F}$ -partition over any field  $\mathbb{F}$ . More precisely, we argue that  $\log \chi_{\mathbb{Z}}(S) \leq O(\text{PPAD}^{\text{cc}}(S))$ . Indeed, let  $\Pi := (V, v^*, o_v, \Pi_v)$  be an optimal PPAD-protocol for  $S$ . We define a  $\mathbb{Z}$ -partition  $\mathcal{M}$  by describing it as a nondeterministic protocol for  $S$  whose accepting computations output weights in  $\mathbb{Z}$  (interpreted as values of the entries of an  $M \in \mathcal{M}$ ): On input  $(x, y)$ , guess a vertex  $v \in V$ ; if  $v$  is a sink in  $G_{x,y}$ , accept with weight 1; if  $v$  is a source distinct from  $v^*$ , accept with weight  $-1$ ; otherwise reject (i.e., weight 0). This protocol accepts with overall weight  $\#(\text{sinks}) - \#(\text{non-distinguished sources}) = 1$  on every input  $(x, y)$ , as desired.

A similar argument yields an analogous query complexity bound  $\text{NS}_{\mathbb{Z}}(F) \leq O(\text{PPAD}^{\text{dt}}(S(F)))$  where  $\text{PPAD}^{\text{dt}}(S)$  is the least cost of a PPAD-*decision tree* (Definition 15) solving  $S$ .

**Nullstellensatz vs. Cutting Planes.** By the above remark,  $F_{\text{bool}}$  for  $F = F_{G,\mathbb{F}}$  admits a low-degree – in fact, constant-degree – Nullstellensatz refutation over any field  $\mathbb{F}$ . Nullstellensatz degree behaves well with respect to compositions: if we compose  $F_{\text{bool}}$  with a gadget  $\text{IND}_m^n$ ,  $m = n^{O(1)}$  (see, e.g., [23, §8] how this can be done), the Nullstellensatz degree can only increase by the query complexity of the gadget, which is  $O(\log n)$  for  $\text{IND}_m^n$ . This gives us an  $n^{O(1)}$ -variate boolean  $k$ -CSP  $F' := F_{\text{bool}} \circ \text{IND}_m^n$  (where  $k$  is constant [23, §8]) such that  $\text{NS}_{\mathbb{F}}(F') \leq O(\log n)$ . On the other hand, we can invoke the strong version of the main result of [23]: if  $F$  has Resolution width  $w$ , then  $F \circ \text{IND}_m^n$  requires Cutting Planes refutations of length  $n^{\Omega(w)}$ . In summary,  $F'$  witnesses that  $\mathbb{F}$ -Nullstellensatz can be exponentially more powerful than log of Cutting Planes length.

### 4.3 Monotone span program lower bounds

Let us prove Theorem 2:  $3\text{LIN}(\mathbb{R})\text{-SAT}_n$  requires exponential-size monotone  $\mathbb{F}_p$ -span programs, that is,

$$\chi_{\mathbb{F}_p}(\text{KW}^+(3\text{LIN}(\mathbb{R})\text{-SAT}_n)) \geq n^{\Omega(1)}. \tag{3}$$

Using Theorem 6 and Lemma 7 similarly as in Subsection 4.2, it suffices to show that  $\text{NS}_{\mathbb{F}_p}(F_{\text{bool}}) \geq n^{\Omega(1)}$ , for some unsatisfiable  $k\text{LIN}(\mathbb{R})$ -CSP  $F$  where  $k$  is a constant. To this end, we consider an  $\mathbb{R}$ -linear system  $F = F_{G,U,\mathbb{R}}$  that generalizes  $F_{G,\mathbb{R}}$  defined above:

$$\forall v \in V : \sum_{(v,u) \in E} z_{(v,u)} - \sum_{(u,v) \in E} z_{(u,v)} = \mathbf{1}_U(v), \tag{F_{G,U,\mathbb{R}}}$$

where  $\mathbb{1}_U: V \rightarrow \{0, 1\}$  is the indicator function for  $U \subseteq V$ . This is unsatisfiable as long as  $U \neq \emptyset$ . Combinatorially, the boolean search problem  $S(F_{\text{bool}})$  can be interpreted as an END-OF- $\ell$ -LINES problem for  $\ell := |U|$ : given a graph with distinguished source vertices  $U$ , find a sink or a source not in  $U$ . It is important to have many distinguished sources,  $|U| \geq n^{\Omega(n)}$ , as otherwise  $S(F_{\text{bool}})$  is in  $\text{PPAD}^{\text{dt}}$  [31] and hence  $F_{\text{bool}}$  has too low an  $\mathbb{F}_p$ -Nullstellensatz degree (by Remark 9).

**Nullstellensatz lower bound.** To show  $\text{NS}_{\mathbb{F}_p}(F_{\text{bool}}) \geq n^{\Omega(1)}$  for an appropriate  $F = F_{G,U,\mathbb{R}}$ , we adapt a result of Beame and Riis [8]. They proved a Nullstellensatz lower bound for a related *bijective pigeonhole* principle  $P_n$  whose underlying graph has *unbounded* degree; we obtain a bounded-degree version of their result by a reduction.

► **Lemma 10** ([8, §8]). *Fix a prime  $p$ . The following system of polynomial equations over variables  $\{x_{ij} : (i, j) \in D \times R\}$ , where  $|D| = n$  and  $|R| = n - n^{\Omega(1)}$ , requires  $\mathbb{F}_p$ -Nullstellensatz degree  $n^{\Omega(1)}$ :*

$$\begin{array}{llll}
(i) & \forall i \in D : & \sum_{j \in R} x_{ij} = 1 & \text{“each pigeon occupies a hole”,} \\
(ii) & \forall j \in R : & \sum_{i \in D} x_{ij} = 1 & \text{“each hole houses a pigeon”,} \\
(iii) & \forall i \in D, \{j, j'\} \in \binom{R}{2} : & x_{ij}x_{ij'} = 0 & \text{“no pigeon occupies two holes”,} \\
(iv) & \forall j \in R, \{i, i'\} \in \binom{D}{2} : & x_{ij}x_{i'j} = 0 & \text{“no hole houses two pigeons”.}
\end{array} \tag{P_n}$$

We construct a natural bounded-degree version  $G$  of the complete bipartite graph  $D \times R$  and show that each constraint of  $F_{\text{bool}}$  for  $F = F_{G,U,\mathbb{R}}$  is a low-degree  $\mathbb{F}_p$ -Nullstellensatz consequence of  $P_n$ . Hence, if  $F_{\text{bool}}$  admits a low-degree  $\mathbb{F}_p$ -Nullstellensatz proof, so does  $P_n$  (see, e.g., [13, Lemma 1] for composing proofs), which contradicts Lemma 10.

The directed graph  $G = (V, E)$  is obtained from the complete bipartite graph  $D \times R$  as illustrated in Figure 2 (for  $|D| = 4$  and  $|R| = 3$ ). Specifically, each vertex of degree  $d$  in  $D \times R$  is replaced with a binary tree of height  $\log d$ . The result is a layered graph with the first and last layers identified with  $D$  and  $R$ , respectively. We also add a “feedback” edge from each vertex in  $R$  to a vertex in  $D$  according to some arbitrary injection  $R \rightarrow D$  (dashed edges in Figure 2). The vertices in  $D$  not incident to feedback edges will form the set  $U$  (singleton in Figure 2).

This defines a boolean 3-CSP  $F_{\text{bool}}$  for  $F = F_{G,U,\mathbb{R}}$  over variables  $\{z_e : e \in E\}$ . In order to reduce  $P_n$  to  $F_{\text{bool}}$ , we define an affine map between the variables  $x_{ij}$  of  $P_n$  and  $z_e$  of  $F_{\text{bool}}$ . Namely, for a feedback edge  $e$  we set  $z_e := 1$ , and for every other  $e = (v, u)$  we set

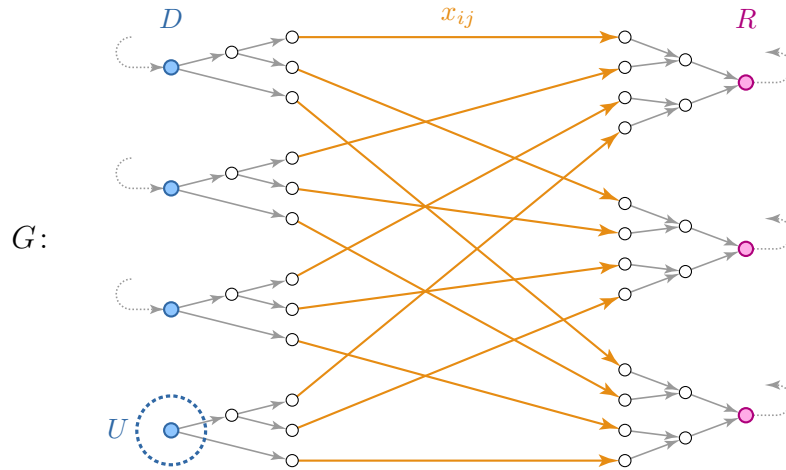
$$z_{(v,u)} := \sum_{i \in D_v} \sum_{j \in R_u} x_{ij},$$

$$\begin{aligned}
\text{where } D_v &:= \{i \in D : v \text{ is reachable from } i \text{ without using feedback edges}\}, \\
R_u &:= \{j \in R : j \text{ is reachable from } u \text{ without using feedback edges}\}.
\end{aligned}$$

Note in particular that this map naturally identifies the edge-variables  $z_e$  in the middle of  $G$  (yellow edges) with the variables  $x_{ij}$  of  $P_n$ . The other variables  $z_e$  are simply affinely dependent on the middle edge-layer. We then show that from the equations of  $P_n$  we can derive each constraint of  $F_{\text{bool}}$ . Recall that the constraint for  $v \in V$  requires that the *out-flow*  $\sum_{(v,u) \in E} z_{(v,u)}$  equals the *in-flow*  $\sum_{(u,v) \in E} z_{(u,v)}$  (plus 1 iff  $v \in U$ ).

$v \notin D \cup R$ : Suppose  $v$  is on the left side of  $G$  (right side is handled similarly) so that  $z_{(v,u)} = \sum_{j \in R_u} x_{ij}$  for some fixed  $i \in D$ . The out-flow is

$$\sum_{(v,u) \in E} z_{(v,u)} = \sum_{(v,u) \in E} \sum_{j \in R_u} x_{ij} = \sum_{j \in R_v} x_{ij}. \tag{4}$$



■ **Figure 2** Graph  $G = (V, E)$ , a bounded-degree version of the biclique  $D \times R$ .

On the other hand,  $v$  has a unique incoming edge  $(u^*, v)$  so the in-flow is  $\sum_{(u,v) \in E} z_{(u,v)} = z_{(u^*,v)} = \sum_{j \in R_v} x_{ij}$ , which equals (4).

$v \in D$ : (Case  $v \in R$  is handled similarly). The in-flow equals 1 (either  $v \in U$  so that we have the +1 term from  $\mathbb{1}_U(v)$ ; or  $v \notin U$  and the value of a feedback-edge variable gives +1).

The out-flow equals  $\sum_{j \in R_v} x_{ij} = \sum_{j \in R} x_{ij} = 1$  by (4),  $R_v = R$ , and (ii).

Finally, we can verify the boolean axioms  $z_e^2 = z_e$ . This holds trivially for feedback edges  $e$ . Let  $e = (v, u)$  be an edge in the left side of  $G$  (right side is similar) so that  $z_e = \sum_{j \in R_u} x_{ij}$  for some fixed  $i \in D$ . We have  $z_e^2 = (\sum_{j \in R_u} x_{ij})^2 = \sum_{j \in R_u} x_{ij}^2 = \sum_{j \in R_u} x_{ij} = z_e$  by (iii) and the boolean axioms for  $P_n$ .

This concludes the reduction and hence the proof of (3).

**PPADS<sup>cc</sup>  $\not\subseteq$  PPA<sup>cc</sup> via End-of- $\ell$ -Lines.** It is straightforward to check that  $F_{\text{bool}}$  for  $F = F_{G,U,\mathbb{R}}$  is in the query class PPADS<sup>dt</sup> (Definition 16). In particular, in the PPADS–decision tree, we can define the distinguished vertex  $v^*$  as being associated with any vertex from  $U$ . Similarly, the lifted problem  $S' := S(F_{\text{bool}}) \circ \text{IND}_n^m$  for  $m = n^{O(1)}$  is in the communication class PPADS<sup>cc</sup>. By contrast, we just proved that  $\chi_{\mathbb{F}_2}(S') \geq n^{\Omega(1)}$ , which implies that  $S' \notin \text{PPA}^{\text{cc}}$ .

## 5 Proofs of Characterizations

Due to space constraints, the proofs of Theorem 3 and Theorem 4 are omitted from this extended abstract. See the full version [24] for complete proofs.

### References

- 1 Michael Alekhovich, Eli Ben-Sasson, Alexander Razborov, and Avi Wigderson. Pseudorandom Generators in Propositional Proof Complexity. *SIAM Journal on Computing*, 34(1):67–88, 2004. doi:10.1137/S0097539701389944.
- 2 László Babai, Peter Frankl, and Janos Simon. Complexity Classes in Communication Complexity Theory. In *Proceedings of the 27th Symposium on Foundations of Computer Science (FOCS)*, pages 337–347. IEEE, 1986. doi:10.1109/SFCS.1986.15.
- 3 László Babai, Anna Gál, and Avi Wigderson. Superpolynomial Lower Bounds for Monotone Span Programs. *Combinatorica*, 19(3):301–319, 1999. doi:10.1007/s004930050058.

- 4 Yakov Babichenko, Shahar Dobzinski, and Noam Nisan. The Communication Complexity of Local Search. Technical report, arXiv, 2018. [arXiv:1804.02676](https://arxiv.org/abs/1804.02676).
- 5 Yakov Babichenko and Aviad Rubinfeld. Communication Complexity of Approximate Nash Equilibria. In *Proceedings of the 49th Symposium on Theory of Computing (STOC)*, pages 878–889. ACM, 2017. doi:10.1145/3055399.3055407.
- 6 Paul Beame, Stephen Cook, Jeff Edmonds, Russell Impagliazzo, and Toniann Pitassi. The Relative Complexity of NP Search Problems. *Journal of Computer and System Sciences*, 57(1):3–19, 1998. doi:10.1006/jcss.1998.1575.
- 7 Paul Beame, Toniann Pitassi, and Nathan Segerlind. Lower Bounds for Lovász–Schrijver Systems and Beyond Follow from Multiparty Communication Complexity. *SIAM Journal on Computing*, 37(3):845–869, 2007. doi:10.1137/060654645.
- 8 Paul Beame and Søren Riis. More on the Relative Strength of Counting Principles. In *Proceedings of the DIMACS Workshop on Proof Complexity and Feasible Arithmetics*, volume 39, pages 13–35, 1998.
- 9 Aleksandrs Belovs, Gábor Ivanyos, Youming Qiao, Miklos Santha, and Siyi Yang. On the Polynomial Parity Argument Complexity of the Combinatorial Nullstellensatz. In *Proceedings of the 32nd Computational Complexity Conference (CCC)*, volume 79, pages 30:1–30:24. Schloss Dagstuhl, 2017. doi:10.4230/LIPIcs.CCC.2017.30.
- 10 Eli Ben-Sasson and Avi Wigderson. Short Proofs Are Narrow—Resolution Made Simple. *Journal of the ACM*, 48(2):149–169, 2001. doi:10.1145/375827.375835.
- 11 Andrei Bulatov. A Dichotomy Theorem for Nonuniform CSPs. In *Proceedings of the 58th Symposium on Foundations of Computer Science (FOCS)*, pages 319–330, 2017. doi:10.1109/FOCS.2017.37.
- 12 Joshua Buresh-Oppenheim and Tsuyoshi Morioka. Relativized NP search problems and propositional proof systems. In *Proceedings of the 19th Conference on Computational Complexity (CCC)*, pages 54–67, 2004. doi:10.1109/CCC.2004.1313795.
- 13 Sam Buss, Dima Grigoriev, Russell Impagliazzo, and Toniann Pitassi. Linear Gaps between Degrees for the Polynomial Calculus Modulo Distinct Primes. *Journal of Computer and System Sciences*, 62(2):267–289, 2001. doi:10.1006/jcss.2000.1726.
- 14 Siu Man Chan. Just a Pebble Game. In *Proceedings of the 28th Conference on Computational Complexity (CCC)*, pages 133–143, 2013. doi:10.1109/CCC.2013.22.
- 15 Siu Man Chan and Aaron Potechin. Tight Bounds for Monotone Switching Networks via Fourier Analysis. *Theory of Computing*, 10(15):389–419, 2014. doi:10.4086/toc.2014.v010a015.
- 16 Stephen Cook, Yuval Filmus, and Dai Tri Man Lê. The Complexity of the Comparator Circuit Value Problem. *ACM Transactions on Computation Theory*, 6(4):15:1–15:44, 2014. doi:10.1145/2635822.
- 17 Constantinos Daskalakis and Christos Papadimitriou. Continuous Local Search. In *Proceedings of the 22nd Symposium on Discrete Algorithms (SODA)*, pages 790–804. SIAM, 2011. URL: <http://dl.acm.org/citation.cfm?id=2133036.2133098>.
- 18 Susanna de Rezende, Jakob Nordström, and Marc Vinyals. How Limited Interaction Hinders Real Communication (and What It Means for Proof and Circuit Complexity). In *Proceedings of the 57th Symposium on Foundations of Computer Science (FOCS)*, pages 295–304. IEEE, 2016. doi:10.1109/FOCS.2016.40.
- 19 John Fearnley, Spencer Gordon, Ruta Mehta, and Rahul Savani. End of Potential Line. Technical report, arXiv, 2018. [arXiv:1804.03450](https://arxiv.org/abs/1804.03450).
- 20 Tomás Feder and Moshe Vardi. The Computational Structure of Monotone Monadic SNP and Constraint Satisfaction: A Study through Datalog and Group Theory. *SIAM Journal on Computing*, 28(1):57–104, 1998. doi:10.1137/S0097539794266766.

- 21 Anna Gál. A Characterization of Span Program Size and Improved Lower Bounds for Monotone Span Programs. *Computational Complexity*, 10(4):277–296, 2001. doi:10.1007/s000370100001.
- 22 Anat Ganor and Karthik C. S. Communication Complexity of Correlated Equilibrium with Small Support. In *Proceedings of the 22nd International Conference on Randomization and Computation (RANDOM)*, volume 116, pages 12:1–12:16. Schloss Dagstuhl, 2018. doi:10.4230/LIPIcs.APPROX-RANDOM.2018.12.
- 23 Ankit Garg, Mika Göös, Pritish Kamath, and Dmitry Sokolov. Monotone Circuit Lower Bounds from Resolution. In *Proceedings of the 50th Symposium on Theory of Computing (STOC)*, pages 902–911. ACM, 2018. doi:10.1145/3188745.3188838.
- 24 Mika Göös, Pritish Kamath, Robert Robere, and Dmitry Sokolov. Adventures in Monotone Complexity and TFNP. Technical Report TR18-163, Electronic Colloquium on Computational Complexity (ECCC), 2018. URL: <https://eccc.weizmann.ac.il/report/2018/163/>.
- 25 Mika Göös and Toniann Pitassi. Communication Lower Bounds via Critical Block Sensitivity. In *Proceedings of the 46th Symposium on Theory of Computing (STOC)*, pages 847–856. ACM, 2014. doi:10.1145/2591796.2591838.
- 26 Mika Göös, Toniann Pitassi, and Thomas Watson. Deterministic Communication vs. Partition Number. In *Proceedings of the 56th Symposium on Foundations of Computer Science (FOCS)*, pages 1077–1088. IEEE, 2015. doi:10.1109/FOCS.2015.70.
- 27 Mika Göös, Toniann Pitassi, and Thomas Watson. The Landscape of Communication Complexity Classes. In *Proceedings of the 43rd International Colloquium on Automata, Languages, and Programming (ICALP)*, pages 86:1–86:15. Schloss Dagstuhl, 2016. doi:10.4230/LIPIcs.ICALP.2016.86.
- 28 Mika Göös and Aviad Rubinfeld. Near-Optimal Communication Lower Bounds for Approximate Nash Equilibria. In *Proceedings of the 59th Symposium on Foundations of Computer Science (FOCS)*, 2018. To appear. arXiv:1805.06387.
- 29 Michelangelo Grigni and Michael Sipser. Monotone Complexity. In *Proceedings of the London Mathematical Society Symposium on Boolean Function Complexity*, pages 57–75. Cambridge University Press, 1992. URL: <http://dl.acm.org/citation.cfm?id=167687.167706>.
- 30 Bernd Halstenberg and Rüdiger Reischuk. Relations Between Communication Complexity Classes. *Journal of Computer and System Sciences*, 41(3):402–429, 1990. doi:10.1016/0022-0000(90)90027-I.
- 31 Alexandros Hollender and Paul Goldberg. The Complexity of Multi-source Variants of the End-of-Line Problem, and the Concise Mutilated Chessboard. Technical report, Electronic Colloquium on Computational Complexity (ECCC), 2018. URL: <https://eccc.weizmann.ac.il/report/2018/120/>.
- 32 Pavel Hubáček and Eylon Yogev. Hardness of Continuous Local Search: Query Complexity and Cryptographic Lower Bounds. In *Proceedings of the 28th Symposium on Discrete Algorithms (SODA)*, pages 1352–1371, 2017. doi:10.1137/1.9781611974782.88.
- 33 Trinh Huynh and Jakob Nordström. On the Virtue of Succinct Proofs: Amplifying Communication Complexity Hardness to Time–Space Trade-Offs in Proof Complexity. In *Proceedings of the 44th Symposium on Theory of Computing (STOC)*, pages 233–248. ACM, 2012. doi:10.1145/2213977.2214000.
- 34 Russell Impagliazzo, Toniann Pitassi, and Alasdair Urquhart. Upper and lower bounds for tree-like cutting planes proofs. In *Proceedings of the 9th Symposium on Logic in Computer Science (LICS)*, pages 220–228. IEEE, 1994. doi:10.1109/LICS.1994.316069.

- 35 David Johnson, Christos Papadimitriou, and Mihalis Yannakakis. How easy is local search? *Journal of Computer and System Sciences*, 37(1):79–100, 1988. doi:10.1016/0022-0000(88)90046-3.
- 36 Stasys Jukna. *Boolean Function Complexity: Advances and Frontiers*, volume 27 of *Algorithms and Combinatorics*. Springer, 2012.
- 37 Mauricio Karchmer and Avi Wigderson. Monotone circuits for connectivity require super-logarithmic depth. In *Proceedings of the 20th Symposium on Theory of Computing (STOC)*, pages 539–550. ACM, 1988. doi:10.1145/62212.62265.
- 38 Mauricio Karchmer and Avi Wigderson. On span programs. In *Proceedings of the 8th Structure in Complexity Theory Conference*, pages 102–111, 1993. doi:10.1109/SCT.1993.336536.
- 39 Eyal Kushilevitz and Noam Nisan. *Communication Complexity*. Cambridge University Press, 1997.
- 40 László Lovász. On determinants, matchings, and random algorithms. In *Proceedings of the 2nd Conference on Fundamentals of Computation Theory (FCT)*, pages 565–574, 1979.
- 41 Ernst Mayr and Ashok Subramanian. The complexity of circuit value and network stability. *Journal of Computer and System Sciences*, 44(2):302–323, 1992. doi:10.1016/0022-0000(92)90024-D.
- 42 Nimrod Megiddo and Christos Papadimitriou. On total functions, existence theorems and computational complexity. *Theoretical Computer Science*, 81(2):317–324, 1991. doi:10.1016/0304-3975(91)90200-L.
- 43 Tsuyoshi Morioka. *Logical Approaches to the Complexity of Search Problems: Proof Complexity, Quantified Propositional Calculus, and Bounded Arithmetic*. PhD thesis, University of Toronto, 2005.
- 44 Ketan Mulmuley. A fast parallel algorithm to compute the rank of a matrix over an arbitrary field. *Combinatorica*, 7(1):101–104, 1987. doi:10.1007/BF02579205.
- 45 Igor Oliveira. *Unconditional Lower Bounds in Complexity Theory*. PhD thesis, Columbia University, 2015. doi:10.7916/D8ZP45KT.
- 46 Christos 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.
- 47 Toniann Pitassi and Robert Robere. Strongly Exponential Lower Bounds for Monotone Computation. In *Proceedings of the 49th Symposium on Theory of Computing (STOC)*, pages 1246–1255. ACM, 2017. doi:10.1145/3055399.3055478.
- 48 Toniann Pitassi and Robert Robere. Lifting Nullstellensatz to Monotone Span Programs over Any Field. In *Proceedings of the 50th Symposium on Theory of Computing (STOC)*, pages 1207–1219. ACM, 2018. doi:10.1145/3188745.3188914.
- 49 Aaron Potechin. Bounds on Monotone Switching Networks for Directed Connectivity. *Journal of the ACM*, 64(4):29:1–29:48, 2017. doi:10.1145/3080520.
- 50 Pavel Pudlák. On extracting computations from propositional proofs (a survey). In *Proceedings of the 30th Foundations of Software Technology and Theoretical Computer Science (FSTTCS)*, volume 8, pages 30–41. Schloss Dagstuhl, 2010. doi:10.4230/LIPIcs.FSTTCS.2010.30.
- 51 Pavel Pudlák and Jiří Sgall. Algebraic models of computation and interpolation for algebraic proof systems. *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, 39:279–295, 1998. doi:10.1090/dimacs/039.
- 52 Ran Raz and Pierre McKenzie. Separation of the Monotone NC Hierarchy. *Combinatorica*, 19(3):403–435, 1999. doi:10.1007/s004930050062.



- 53 Alexander Razborov. Lower bounds on monotone complexity of the logical permanent. *Mathematical notes of the Academy of Sciences of the USSR*, 37(6):485–493, 1985. doi:10.1007/BF01157687.
- 54 Alexander Razborov. Lower bounds on the monotone complexity of some Boolean functions. *Doklady Akademii Nauk USSR*, 285:798–801, 1985.
- 55 Alexander Razborov. Unprovability of lower bounds on circuit size in certain fragments of bounded arithmetic. *Izvestiya of the RAN*, pages 201–224, 1995.
- 56 Robert Robere, Toniann Pitassi, Benjamin Rossman, and Stephen Cook. Exponential Lower Bounds for Monotone Span Programs. In *Proceedings of the 57th Symposium on Foundations of Computer Science (FOCS)*, pages 406–415. IEEE, 2016. doi:10.1109/FOCS.2016.51.
- 57 Tim Roughgarden. Complexity Theory, Game Theory, and Economics. Technical report, arXiv, 2018. arXiv:1801.00734.
- 58 Tim Roughgarden and Omri Weinstein. On the Communication Complexity of Approximate Fixed Points. In *Proceedings of the 57th Symposium on Foundations of Computer Science (FOCS)*, pages 229–238. IEEE, 2016. doi:10.1109/FOCS.2016.32.
- 59 Thomas Schaefer. The Complexity of Satisfiability Problems. In *Proceedings of the 10th Symposium on Theory of Computing (STOC)*, pages 216–226. ACM, 1978. doi:10.1145/800133.804350.
- 60 Dmitry Sokolov. Dag-Like Communication and Its Applications. In *Proceedings of the 12th Computer Science Symposium in Russia (CSR)*, pages 294–307. Springer, 2017. doi:10.1007/978-3-319-58747-9\_26.
- 61 Éva Tardos. The gap between monotone and non-monotone circuit complexity is exponential. *Combinatorica*, 8(1):141–142, 1988. doi:10.1007/BF02122563.
- 62 Alasdair Urquhart. Hard examples for resolution. *Journal of the ACM*, 34(1):209–219, 1987. doi:10.1145/7531.8928.
- 63 Dmitriy Zhuk. A Proof of CSP Dichotomy Conjecture. In *Proceedings of the 58th Symposium on Foundations of Computer Science (FOCS)*, pages 331–342, 2017. doi:10.1109/FOCS.2017.38.

## A Appendix: TFNP Class Definitions

For each TFNP subclass there is a canonical definition of its communication or query analogue: we simply let communication protocols or decision trees (rather than circuits) implicitly define the objects that appear in the original Turing machine definition. Each communication class  $C^{cc}$  (resp. query class  $C^{dt}$ ) is defined via a  $C$ -protocol (resp.  $C$ -decision tree) that solves a two-party search problem  $S \subseteq \{0, 1\}^{n/2} \times \{0, 1\}^{n/2} \times \mathcal{O}$  (resp.  $S \subseteq \{0, 1\}^n \times \mathcal{O}$ ). The class  $C^{cc}$  (resp.  $C^{dt}$ ) is then defined as the set of all  $n$ -bit search problems  $S$  that admit a polylog( $n$ )-cost  $C$ -protocol (resp.  $C$ -decision tree). We only define the communication analogues below with the understanding that a query version can be obtained by replacing mentions of a protocol  $\Pi_v(x, y)$  by a decision tree  $\mathcal{T}_v(x)$ ; the *cost* of a  $C$ -decision tree is defined as  $\max_{v,x} \#(\text{queries made by } \mathcal{T}_v(x))$ . In what follows, *sink* means out-degree 0, and *source* means in-degree 0.

### ► Definition 11. (FP)

**Syntax:**  $\Pi$  is a (deterministic) protocol outputting values in  $\mathcal{O}$ .

**Object:**  $n/a$

**Correctness:**  $\Pi(x, y) \in S(x, y)$ .

**Cost:**  $|\Pi| :=$  communication cost of  $\Pi$ .

► **Definition 12.** (EoML)

**Syntax:**  $V$  is a vertex set with a distinguished vertex  $v^* \in V$ . For each  $v \in V$ :  $o_v \in \mathcal{O}$  and

$\Pi_v$  is a protocol outputting a tuple  $(s_v(x, y), p_v(x, y), \ell_v(x, y)) \in V \times V \times \mathbb{Z}$ .

**Object:** Dag  $G_{x,y} = (V, E)$  where  $(v, u) \in E$  iff  $s_v(x, y) = u$ ,  $p_u(x, y) = v$ ,  $\ell_v(x, y) > \ell_u(x, y)$ .

**Correctness:** If  $v^*$  is a sink or non-source in  $G_{x,y}$ , then  $o_{v^*} \in S(x, y)$ .

If  $v \neq v^*$  is a sink or source in  $G_{x,y}$ , then  $o_v \in S(x, y)$ .

**Cost:**  $\log |V| + \max_v |\Pi_v|$ .

► **Definition 13.** (SoML)

**Syntax:** Same as in Definition 12.

**Object:** Same as in Definition 12.

**Correctness:** If  $v^*$  is a sink or non-source in  $G_{x,y}$ , then  $o_{v^*} \in S(x, y)$ .

If  $v \neq v^*$  is a sink in  $G_{x,y}$ , then  $o_v \in S(x, y)$ .

**Cost:**  $\log |V| + \max_v |\Pi_v|$ .

► **Definition 14.** (PLS)

**Syntax:**  $V$  is a vertex set. For each  $v \in V$ :  $o_v \in \mathcal{O}$  and  $\Pi_v$  is a protocol outputting a pair

$(s_v(x, y), \ell_v(x, y)) \in V \times \mathbb{Z}$ .

**Object:** Dag  $G_{x,y} = (V, E)$  where  $(v, u) \in E$  iff  $s_v(x, y) = u$  and  $\ell_v(x, y) > \ell_u(x, y)$ .

**Correctness:** If  $v$  is a sink in  $G_{x,y}$ , then  $o_v \in S(x, y)$ .

**Cost:**  $\log |V| + \max_v |\Pi_v|$ .

► **Definition 15.** (PPAD)

**Syntax:**  $V$  is a vertex set with a distinguished vertex  $v^* \in V$ . For each  $v \in V$ :  $o_v \in \mathcal{O}$  and

$\Pi_v$  is a protocol outputting a pair  $(s_v(x, y), p_v(x, y)) \in V \times V$ .

**Object:** Digraph  $G_{x,y} = (V, E)$  where  $(v, u) \in E$  iff  $s_v(x, y) = u$  and  $p_u(x, y) = v$ .

**Correctness:** If  $v^*$  is a sink or non-source in  $G_{x,y}$ , then  $o_{v^*} \in S(x, y)$ .

If  $v \neq v^*$  is a sink or source in  $G_{x,y}$ , then  $o_v \in S(x, y)$ .

**Cost:**  $\log |V| + \max_v |\Pi_v|$ .

► **Definition 16.** (PPADS)

**Syntax:** Same as in Definition 15.

**Object:** Same as in Definition 15.

**Correctness:** If  $v^*$  is a sink or non-source in  $G_{x,y}$ , then  $o_{v^*} \in S(x, y)$ .

If  $v \neq v^*$  is a sink in  $G_{x,y}$ , then  $o_v \in S(x, y)$ .

**Cost:**  $\log |V| + \max_v |\Pi_v|$ .

► **Definition 17.** (PPA)

**Syntax:**  $V$  is a vertex set with a distinguished vertex  $v^* \in V$ . For each  $v \in V$ :  $o_v \in \mathcal{O}$  and

$\Pi_v$  is a protocol outputting a subset  $\Pi_v(x, y) \subseteq V$  of size at most 2.

**Object:** Undirected graph  $G_{x,y} = (V, E)$  where  $\{v, u\} \in E$  iff  $v \in \Pi_u(x, y)$  and  $u \in \Pi_v(x, y)$ .

**Correctness:** If  $v^*$  has degree  $\neq 1$  in  $G_{x,y}$ , then  $o_{v^*} \in S(x, y)$ .

If  $v \neq v^*$  has degree  $\neq 2$  in  $G_{x,y}$ , then  $o_v \in S(x, y)$ .

**Cost:**  $\log |V| + \max_v |\Pi_v|$ .

► **Definition 18.** (PPP)

**Syntax:**  $V$  is a vertex set with a distinguished vertex  $v^* \in V$ . For each unordered pair

$\{v, u\} \in \binom{V}{2}$ :  $o_{\{v,u\}} \in \mathcal{O}$ . For each  $v \in V$ :  $\Pi_v$  is a protocol outputting values in  $V - v^*$ .

**Object:** Bipartite graph  $G_{x,y} = (V \times (V - v^*), E)$  where  $(v, w) \in E$  iff  $\Pi_v(x, y) = w$ .

**Correctness:** If  $(v, w)$  and  $(u, w)$ ,  $v \neq u$ , are edges in  $G_{x,y}$ , then  $o_{\{v,u\}} \in S(x, y)$ .

**Cost:**  $\log |V| + \max_v |\Pi_v|$ .



# Algorithmic Polarization for Hidden Markov Models

Venkatesan Guruswami<sup>1</sup>

Computer Science Department, Carnegie Mellon University, Pittsburgh, PA 15213, USA  
venkatg@cs.cmu.edu

Preetum Nakkiran<sup>2</sup>

Harvard John A. Paulson School of Engineering and Applied Sciences, Harvard University,  
33 Oxford Street, Cambridge, MA 02138, USA  
preetum@cs.harvard.edu

Madhu Sudan<sup>3</sup>

Harvard John A. Paulson School of Engineering and Applied Sciences, Harvard University,  
33 Oxford Street, Cambridge, MA 02138, USA  
madhu@cs.harvard.edu

---

## Abstract

Using a mild variant of polar codes we design linear compression schemes compressing Hidden Markov sources (where the source is a Markov chain, but whose state is not necessarily observable from its output), and to decode from Hidden Markov channels (where the channel has a state and the error introduced depends on the state). We give the first polynomial time algorithms that manage to compress and decompress (or encode and decode) at input lengths that are polynomial *both* in the gap to capacity and the mixing time of the Markov chain. Prior work achieved capacity only asymptotically in the limit of large lengths, and polynomial bounds were not available with respect to either the gap to capacity or mixing time. Our results operate in the setting where the source (or the channel) is *known*. If the source is *unknown* then compression at such short lengths would lead to effective algorithms for learning parity with noise – thus our results are the first to suggest a separation between the complexity of the problem when the source is known versus when it is unknown.

**2012 ACM Subject Classification** Theory of computation → Error-correcting codes

**Keywords and phrases** polar codes, error-correcting codes, compression, hidden markov model

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2019.39

## 1 Introduction

We study the problem of designing coding schemes, specifically encoding and decoding algorithms, that overcome errors caused by stochastic, but not memoryless, channels. Specifically we consider the class of “(hidden) Markov channels” that are stateful, with the states evolving according to some Markov process, and where the distribution of error depends on

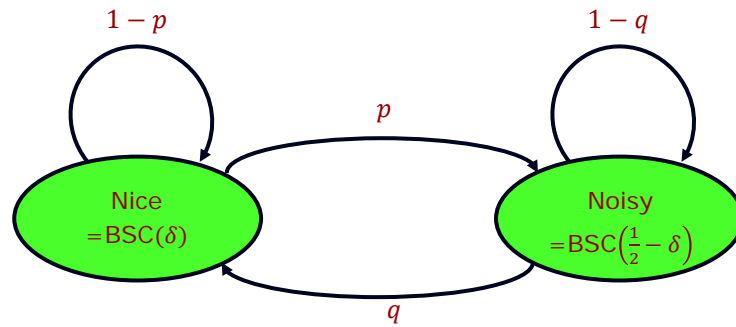
---

<sup>1</sup> Most of this work was done when the author was visiting the Center for Mathematical Sciences and Applications, Harvard University, Cambridge, MA. Research supported in part by NSF grants CCF-1422045 and CCF-1814603.

<sup>2</sup> Work supported in part by the NSF Graduate Research Fellowship Grant No. DGE1144152, and Madhu Sudan’s Simons Investigator Award and NSF Award CCF 1715187.

<sup>3</sup> Work supported in part by a Simons Investigator Award and NSF Award CCF 1715187.





■ **Figure 1** A Markovian Channel: The Nice state flips bits with probability  $\delta$  whereas the Noisy state flips with probability  $1/2 - \delta$ . The stationary probability of the Nice state is  $q/p$  times that of the Noisy state.

the state.<sup>4</sup> Such Markovian models capture many natural settings of error, such as bursty error models. (See for example, Figure 1.) Yet they are often less understood than their memoryless counterparts (or even “explicit Markov models” where the state is completely determined by the actions of the channel). For instance (though this is not relevant to our work) even the capacity of such channels is not known to have a closed form expression in terms of channel parameters. (In particular the exact capacity of the channel in Figure 1 is not known as a function of  $\delta$ ,  $p$  and  $q$ !)

In this work we aim to design coding schemes that achieve rates arbitrarily close to capacity. Specifically given a channel of capacity  $C$  and gap parameter  $\varepsilon > 0$ , we would like to design codes that achieve a rate of at least  $C - \varepsilon$ , that admit polynomial time algorithms even at small block lengths  $n \geq \text{poly}(1/\varepsilon)$ . Even for the memoryless case such coding schemes were not known till recently. In 2008, Arikan [1] invented a completely novel approach to constructing codes based on “channel polarization” for communication on binary-input memoryless channels, and proved that they enable achieving capacity in the limit of large code lengths with near-linear complexity encoding and decoding. In 2013, independent works by Guruswami and Xia [5] and Hassani et al. [6] gave a finite-length analysis of Arikan’s polar codes, proving that they approach capacity fast, at block lengths bounded by  $\text{poly}(1/\varepsilon)$  where  $\varepsilon > 0$  is the difference between the channel capacity and code rate.

The success of polar codes on the memoryless channels might lead to the hope that maybe these codes, or some variants, might lead to similar coding schemes for channels with memory. But such a hope is not easily justified: the analysis of polar codes relies heavily on the fact that errors introduced by the channel are independent and this is exactly what is not true for channels with memory. Despite this seemingly insurmountable barrier, Şaşıoğlu [4] and later Şaşıoğlu and Tal [9] showed, quite surprisingly, that the analysis of polar codes can be carried out even with Markovian channels (and potentially even broader classes of channels). Specifically they show that these codes converge to capacity and even the probability of decoding error, under maximum likelihood decoding, drops exponentially fast in the block length (specifically as  $2^{-n^{\Omega(1)}}$  on codes of length  $n$ ; see also [10], where exponentially fast polarization was also shown at the high entropy end). An extension of Arikan’s successive cancellation decoder from the memoryless case was also given by [12], building on an earlier version [13] specific to intersymbol interference channels, leading to efficient decoding algorithms.

<sup>4</sup> We use the term *hidden* to emphasize the fact that the state itself is not directly observable from the actions of the channel, though in the interest of succinctness we will omit this term for most of the rest of this section.

However, none of the works above give small bounds on the block length of the codes as a function of the gap to capacity, and more centrally to this work, on the mixing time of the Markov chain. The latter issue gains importance when we turn to the issue of “compressing Markov sources” which turns out to be an intimately related task to that of error-correction for Markov channels as we elaborate below and which is also the central task we turn to in this paper. We start by describing Markov source and the (linear) compression problem.

A (hidden) Markov source over alphabet  $\Sigma$  is given by a Markov chain on some finite state space where each state  $s$  has an associated distribution  $D_s$  over  $\Sigma$ . The source produces information by performing a walk on the chain and at each time step  $t$ , outputting a letter of  $\Sigma$  drawn according to the distribution associated with the state at time  $t$  (independent of all previous choices, and previous states).<sup>5</sup> In the special case of additive Markovian channels where the output of the channel is the sum of the transmitted word with an error vector produced by a Markov source, a well-known correspondence shows that error-correction for the additive Markov channel reduces to the task of designing a compression and decompression algorithm for Markovian sources, with the compression being *linear*. Indeed in this paper we only focus on this task: our goal turns into that of compressing  $n$  bits generated by the source to its entropy upto an additive factor of  $\varepsilon n$ , while  $n$  is only polynomially large in  $1/\varepsilon$ .

A central issue in the task of compressing a source is whether the source is *known* to the compression algorithm or not. While ostensibly the problem should be easier in the “known” setting than in the “unknown” one, we are not aware of any formal results suggesting a difference in complexity. It turns out that compression in the setting where the source is unknown is at least as hard as “learning parity with noise” (we argue this in Appendix B), *if* the compression works at lengths polynomial in the mixing time and gap to capacity. This suggests that the unknown source setting is hard (under some current beliefs). No corresponding hardness was known for the task of compressing sources when they are known, but no easiness result seems to have been known either (and certainly no linear compression algorithm was known). This leads to the main question addressed (positively) in this work.

## Our Results

Our main result is a construction of codes for *additive Markov channels* that gets  $\varepsilon$  close to capacity at block lengths polynomial in  $1/\varepsilon$  and the mixing time of the Markov chain, with polynomial (in fact near-linear) encoding and decoding time. Informally additive channels are those that map inputs from some alphabet  $\Sigma$  to outputs over  $\Sigma$  with an abelian group defined on  $\Sigma$  and the channel generates an error sequence independent of the input sequence, and the output of the channel is just the coordinatewise sum of the input sequence with the error sequence. (In our case the alphabet  $\Sigma$  is a finite field of prime cardinality.) The exact class of channels is described in Definition 4, and Theorem 10 states our result formally. We stress that we work with additive channels only for conceptual simplicity and that our results should extend to more general symmetric channels though we don’t do so here. Prior to this work no non-trivial Markov channel was known to achieve efficient encoding and decoding at block lengths polynomial in either parameter (gap to capacity or mixing time).

Our construction and analyses turn out to be relatively simple given the works of Şaşıoğlu and Tal [4, 9] and the work of Blasiok et al. [2]. The former provides insights on how to work with channels with memory, whereas the latter provides tools needed to get short block

---

<sup>5</sup> The phrase “hidden” emphasizes the fact that the output produced by the source does not necessarily reveal the sequence of states visited.

length and cleaner abstractions of the efficient decoding algorithm that enable us to apply it in our setting. Our codes are a slight variant of polar codes, where we apply the polar transforms independently to blocks of inputs. This enables us to apply the analysis of [2] in an essentially black box manner, benefiting both from its polynomially fast convergence guarantee to capacity as well as its generality covering all polarizing matrices over any prime alphabet (and not just the basic Boolean  $2 \times 2$  transform covered in [9]).

We give a more detailed summary of how our codes are obtained and how we analyze them in Section 3 after stating our results and main theorem formally.

## 2 Definitions and Main Results

### 2.1 Notation and Definitions

We will use  $\mathbb{F}_q$  to denote the finite field with  $q$  elements. Throughout the paper, we will deal only with the case when  $q$  is a prime. (This restriction in turn comes from the work of [2] whose results we use here.)

We use several notations to index matrices. For a matrix  $M \in \mathbb{F}_q^{m \times n}$ , the entry in the  $i$ th row,  $j$ th column is denoted  $M_{i,j}$  or  $M_{(i,j)}$ . Columns are denoted by superscripts, i.e.,  $M^j \in \mathbb{F}_q^m$  denotes the  $j$ th column of  $M$ . Note that  $M_i^j = M_{(i,j)}$ . We also use the indices as sets in the natural way. For example  $M^{\leq j} \in \mathbb{F}_q^{m \times j}$  denotes the first  $j$  columns of  $M$ .  $M_{\leq i}^{\leq j}$  denotes the submatrix of elements in the first  $j$  columns and first  $i$  rows.  $M_{\prec(i,j)}$  denotes the set of elements of  $M$  indexed by lexicographically smaller indices than  $(i, j)$ . Multiplication of a matrix  $M \in \mathbb{F}_q^{m \times n}$  with a vector  $v \in \mathbb{F}_q^n$  is denoted  $Mv$ .

For a finite set  $S$ , let  $\Delta(S)$  denote the set of probability distributions over  $S$ . For a random variable  $X$  and event  $E$ , we write  $X|E$  to denote the conditional distribution of  $X$ , conditioned on  $E$ . For example, we may write  $X|\{X_1 = 0\}$ .

The *total-variation distance* between two distributions  $p, q \in \Delta(U)$  is

$$\|p - q\|_1 := \sum_i |p(i) - q(i)|$$

We consider compression schemes, as a map  $\mathbb{F}_q^n \rightarrow \mathbb{F}_q^m$ . The *rate* of a compression scheme  $\mathbb{F}_q^n \rightarrow \mathbb{F}_q^m$  is the ratio  $m/n$ .

For a random variable  $X \in [q]$ , the (*non-normalized*) *entropy* is denoted  $H(X)$ , and is

$$H(X) := - \sum_i \Pr[X = i] \log(\Pr[X = i])$$

and the *normalized entropy* is denoted  $\bar{H}(X)$ , and is

$$\bar{H}(X) := \frac{1}{\log(q)} H(X)$$

► **Definition 1.** A *Markov chain*  $\mathcal{M} = (\ell, \Pi, \pi_0)$  is given by an  $\ell$  representing the state space  $[\ell]$ , a transition matrix  $\Pi \in \mathbb{R}^{\ell \times \ell}$ , and a distribution on initial state  $\pi_0 \in \Delta([\ell])$ . The rows of  $\Pi$ , denoted  $\Pi_1, \dots, \Pi_\ell$  are thus elements of  $\Delta([\ell])$ . A Markov chain generates a random sequence of states  $X_0, X_1, X_2, \dots$  determined by letting  $X_0 \sim \pi_0$ , and  $X_t \sim \Pi_{X_{t-1}}$  for  $t > 0$  given  $X_0, \dots, X_{t-1}$ . The stationary distribution  $\pi \in \Delta([\ell])$  is the distribution such that if  $X_0 \sim \pi$ , then all  $X_t$ 's are marginally identically distributed as  $\pi$ .

We consider only Markov chains which are irreducible and aperiodic, and hence have a stationary distribution to which they converge in the limit. The rate of convergence is measured by the mixing time, defined below.



► **Definition 2.** The *mixing time* of a Markov chain is the constant  $\tau > 0$  such that for every initial state  $s_0$  of the Markov chain, the distribution of state  $s_\ell$  is  $\exp(-\ell/\tau)$ -close in total variation distance to the stationary distribution  $\pi$ .

► **Definition 3.** A *(stationary, hidden) Markov source*  $\mathcal{H} = (\Sigma, \mathcal{M}, \{\mathcal{S}_1, \dots, \mathcal{S}_\ell\})$  is specified by an alphabet  $\Sigma$ , a Markov chain  $\mathcal{M}$  on  $\ell$  states and distributions  $\{\mathcal{S}_i \in \Delta(\Sigma)\}_{i \in [\ell]}$ . The output of the source is a sequence  $Z_1, Z_2, \dots$ , of random variables obtained by first sampling a sequence  $X_0, X_1, X_2, \dots$  according to  $\mathcal{M}$  and then sampling  $Z_i \sim \mathcal{S}_{X_i}$  independently for each  $i$ . We let  $\mathcal{H}_t$  the distribution of output sequences of length  $t$ , and  $\mathcal{H}_t^{\otimes s}$  denote the distribution of  $s$  i.i.d. samples from  $\mathcal{H}_t$ .

Similarly, we define an *additive Markov channel* as a channel which adds noise from a Markov source.

► **Definition 4.** An *additive Markov channel*  $\mathcal{C}_{\mathcal{H}}$ , specified by a Markov source  $\mathcal{H}$  over alphabet  $\mathbb{F}_q$ , is a randomized map  $\mathcal{C}_{\mathcal{H}} : \mathbb{F}_q^* \rightarrow \mathbb{F}_q^*$  obtained as follows: On channel input  $X_1, \dots, X_n$ , the channel outputs  $Y_1, \dots, Y_n$  where  $Y_i = X_i + Z_i$  where  $Z = (Z_1, \dots, Z_n) \sim \mathcal{H}_n$ .

► **Definition 5.** A *linear code* is a linear map  $C : \mathbb{F}_q^k \rightarrow \mathbb{F}_q^n$ . The *rate* of a code is the ratio  $k/n$ .

► **Definition 6.** For all sets  $A, B$ , a *constructive source over  $(A|B)$  samplable in time  $T$*  is a distribution  $\mathcal{D} \in \Delta(A \times B)$  such that  $(a, b) \sim \mathcal{D}$  can be sampled efficiently in time at most  $T$ , and for every fixed  $b \in B$ , the conditional distribution  $A|\{B = b\}$  can be sampled efficiently in time at most  $T$ .

► **Proposition 7.** *Every Markov source with state space  $[\ell]$  is a constructive source samplable in time  $\mathcal{O}(n\ell^2)$ . That is, for every  $n$ , let  $Y_1, \dots, Y_n$  be the random variables generated by the Markov source. Then, the sequence  $Y_1, \dots, Y_n$  can be sampled in time at most  $\mathcal{O}(n\ell^2)$ , and moreover for every setting of  $Y_{<n} = y_{<n}$ , the distribution  $(Y_n|Y_{<n} = y_{<n})$  can be sampled in time  $\mathcal{O}(n\ell^2)$ .*

**Proof.** Sampling  $Y_1, \dots, Y_n$  can clearly be done by simulating the Markov chain, and sampling from the conditional distribution  $(Y_n|Y_{<n} = y_{<n})$  is possible using the standard *Forward Algorithm* for inference in Hidden Markov Models, which we describe for completeness in Appendix A. ◀

Finally, we will use the following notion of *mixing matrices* from [7, 2], characterizing which matrices lead to good polar codes. In the study of polarization it is well-known that lower-triangular matrices do not polarize at all, and the polarization characteristics of matrices are invariant under column permutations. Mixing matrices are defined to be those that avoid the above cases.

► **Definition 8.** For prime  $q$  and  $M \in \mathbb{F}_q^{k \times k}$ ,  $M$  is said to be a *mixing matrix* if  $M$  is invertible and for every permutation of the columns of  $M$ , the resulting matrix is not lower-triangular.

## 2.2 Main Theorems

We are now ready to state the main results of this work formally. We begin with the statement for compressing the output of a hidden Markov model.

► **Theorem 9.** *For every prime  $q$  and mixing matrix  $M \in \mathbb{F}_q^{k \times k}$  there exists a preprocessing algorithm (POLAR-PREPROCESS, Algorithm 6.3), a compression algorithm (POLAR-COMPRESS, Algorithm 4.1), a decompression algorithm (POLAR-DECOMPRESS, Algorithm 4.2) and a polynomial  $p(\cdot)$  such that for every  $\varepsilon > 0$ , the following properties hold:*

1. POLAR-PREPROCESS is a randomized algorithm that takes as input a Markov source  $\mathcal{H}$  with  $\ell$  states, and  $t \in \mathbb{N}$ , and runs in time  $\text{poly}(n, \ell, 1/\varepsilon, q)$  where  $n = k^{2t}$  and outputs auxiliary information for the compressor and decompressor (for  $\mathcal{H}_n$ ).
2. POLAR-COMPRESS takes as input a sequence  $Z \in \mathbb{F}_q^n$  as well as the auxiliary information output by the preprocessor, runs in time  $\mathcal{O}(n \log n)$ , and outputs a compressed string  $\tilde{U} \in \mathbb{F}_q^{\overline{H}(Z)+\varepsilon n}$ . Further, for every auxiliary input, the map  $Z \rightarrow \tilde{U}$  is a linear map.
3. POLAR-DECOMPRESS takes as input a Markov source  $\mathcal{H}$  a compressed string  $\tilde{U} \in \mathbb{F}_q^{\overline{H}(Z)+\varepsilon n}$  and the auxiliary information output by the preprocessor, runs in time  $\mathcal{O}(n^{3/2}\ell^2 + n \log n)$  and outputs  $\hat{Z} \in \mathbb{F}_q^n$ .<sup>6</sup>

The guarantee provided by the above algorithms is that with probability at least  $1 - \exp(-\Omega(n))$ , the Preprocessing Algorithm outputs auxiliary information  $S$  such that

$$\Pr_{Z \sim \mathcal{H}_n} [\text{POLAR-DECOMPRESS}(\mathcal{H}, S; \text{POLAR-COMPRESS}(Z; S)) \neq Z] \leq \mathcal{O}\left(\frac{1}{n^2}\right),$$

provided  $n > p(\tau/\varepsilon)$  where  $\tau$  is the mixing time of  $\mathcal{H}$ .

(In the above  $\mathcal{O}(\cdot)$  hides constants depending  $k$  and  $q$ , but not on  $\ell$  or  $n$ .)

The above linear compression directly yields channel coding for additive Markov channels, via a standard reduction (the details of which are in Section 7.)

► **Theorem 10.** For every prime  $q$  and mixing matrix  $M \in \mathbb{F}_q^{k \times k}$  there exists a randomized preprocessing algorithm PREPROCESS, an encoding algorithm ENC, a decoding algorithm DEC, and a polynomial  $p(\cdot)$  such that for every  $\varepsilon > 0$ , the following properties hold:

1. PREPROCESS is a randomized algorithm that takes as input an additive Markov channel  $\mathcal{C}_{\mathcal{H}}$  described by Markov source  $\mathcal{H}$  with  $\ell$  states, and  $t \in \mathbb{N}$ , and runs in time  $\text{poly}(n, \ell, 1/\varepsilon)$  where  $n = k^{2t}$ , and outputs auxiliary information for  $\mathcal{H}_n$ .
2. ENC takes as input a message  $x \in \mathbb{F}_q^r$ , where  $r \geq n(1 - \frac{\overline{H}(Z)}{n} - \varepsilon)$ , as well as auxiliary information from the preprocessor and outputs and computes  $\text{ENC}(x) \in \mathbb{F}_q^n$  in  $\mathcal{O}(n \log n)$  time.
3. DEC takes as input the Markov source  $\mathcal{H}$ , auxiliary information from the preprocessor and a string  $z \in \mathbb{F}_q^n$ , runs in time  $\mathcal{O}_q(n^{3/2}\ell^2 + n \log n)$ , and outputs an estimate  $\hat{x} \in \mathbb{F}_q^r$  of the message  $x$ .<sup>7</sup>

The guarantee provided by the above algorithms is that with probability at least  $1 - \exp(-\Omega(n))$ , the Preprocessing algorithm outputs  $S$  such that for all  $x \in \mathbb{F}_q^r$  we have

$$\Pr_{\mathcal{C}_{\mathcal{H}}} [\text{DEC}(\mathcal{H}; \mathcal{C}_{\mathcal{H}}(\text{ENC}(C; x))) \neq x] \leq \mathcal{O}\left(\frac{1}{n^2}\right),$$

provided  $n > p(\tau/\varepsilon)$  where  $\tau$  is the mixing time of  $\mathcal{H}$ .

(In the above  $\mathcal{O}(\cdot)$  hides constants that may depend on  $k$  and  $q$  but not on  $\ell$  or  $n$ .)

Theorem 10 follows relatively easily from Theorem 9 and so in the next section we focus on the overview of the proof of the latter.

<sup>6</sup> The runtime of the decompression algorithm can be improved to a runtime of  $\mathcal{O}(n^{1+\delta}\ell^2 + n \log n)$  by a simple modification. In particular, by taking the input matrix  $Z$  to be  $n^{1-\delta} \times n^\delta$  instead of  $n^{1/2} \times n^{1/2}$ . In fact we believe the decoding algorithm can be improved to an  $\mathcal{O}(n \log n)$  time algorithm with some extra bookkeeping though we don't do so here.

<sup>7</sup> This can similarly be improved to a runtime of  $\mathcal{O}_q(n^{1+\delta}\ell^2 + n \log n)$ .

### 3 Overview of our construction

**Basics of polarization.** We start with the basics of polarization in the setting of compressing samples from an i.i.d. source. To compress a sequence  $Z \in \mathbb{F}_2^n$  drawn from some source, the idea is to build an invertible linear function  $P$  such that for all but  $\varepsilon$  fraction of the output coordinates  $i \in [n]$ , the conditional entropy  $H(P(Z)_i | P(Z)_{<i})$  is close to 0 and or close to 1. (Such an effect is called *polarization*, as the entropies are driven to polarize toward the two extreme values.) Since a deterministic invertible transformation preserves the total entropy, it follows that roughly  $H(Z)$  output coordinates can have entropy close to 1 and  $n - H(Z)$  coordinates have (conditional) entropy close to 0. Letting  $S$  denote the coordinates whose conditional entropies that are not close to zero, the compression function is simply  $Z \mapsto P(Z)_S$ , the projection of the output  $P(Z)$  onto the coordinates in  $S$ .

Picking a random linear function  $P$  would satisfy the properties above with high probability, but this is not known (and unlikely) to be accompanied by efficient algorithms. To get the algorithmics (how to compute  $P$  efficiently, to determine  $S$  efficiently, and to decompress efficiently) one uses a recursive construction of  $P$ . For our purposes the following explanation works best: Let  $n = m^2$  and view  $Z = (Z_{11}, Z_{12}, \dots, Z_{mm})$  and as an  $m \times m$  matrix over  $\mathbb{F}_2$ , where the elements of  $Z$  arrive one row at a time. Let  $P_m^{\text{row}}(\cdot)$  denote the operation mapping  $\mathbb{F}_2^{m \times m}$  to  $\mathbb{F}_2^{m \times m}$  that applies  $P_m$  to each row of separately. Let  $P_m^{\text{column}}(\cdot)$  denote the operation that applies  $P_m$  to each column separately. Then  $P_n(Z) = P_m^{\text{column}}(P_m^{\text{row}}(Z))^T$ . The base case is given by  $P_2(U, V) = (U + V, V)$ .

Intuitively, when the elements of  $Z$  are *independent* and identical, the operation  $P_m$  already polarizes the outputs somewhat and so a moderate fraction of the outputs of  $P_m^{\text{row}}(Z)$  have conditional entropies moderately close to 0 or 1. The further application of  $P_m^{\text{column}}(\cdot)$  further polarizes the output bringing a larger fraction of the conditional entropies of the output even closer to 0 or 1.

**Polarization for Markovian Sources.** When applied to source  $Z$  with memory, roughly the analysis in [9], reinterpreted to facilitate our subsequent modification of the above polar construction, goes as follows: Since the elements of the row  $Z_i$  are not really independent one cannot count on the polarization effects of  $P_m^{\text{row}}$ . But, letting  $U = P_m^{\text{row}}(Z)$  one can show that most elements of the column of  $U^j$  are almost independent of each other, provided  $m$  is much larger than the mixing time of the source. (Here we imagine that the entries of  $Z$  arrive row-by-row, so that the source outputs within each row are temporally well-separated from most entries of the previous row, when  $m$  is large.) Further, this almost independence holds even when conditioning on the columns  $U^{<j}$  for most values of  $j$ . Thus the operation  $P_m^{\text{column}}(\cdot)$  continues to have its polarization effects and this is good enough to get a qualitatively strong polarization theorem (about the operator  $P_n$ !).

The above analysis is asymptotic, proving that in the limit of  $n \rightarrow \infty$ , we get optimal compression. However, we do not know how to give an effective finite-length analysis of the polarization process for Markovian process, as the analysis in [5, 6] crucially rely on independence which we lack within a row.

**Our Modified Code and Ingredients of Analysis.** To enable a finite-length analysis, we make a minor, but quite important, alteration to the polar code: Instead of using  $P_n(Z) = P_m^{\text{column}}(P_m^{\text{row}}(Z))^T$  we simply use the transformation  $\tilde{P}_n = P_m^{\text{column}}(Z)^T$  (or in other words, we replace the inner function  $P_m^{\text{row}}(\cdot)$  in the definition of  $P_n$  by the identity function). This implies that we lose whatever polarization effects of  $P_m^{\text{row}}$  we may have been counting on, but as pointed out above, for Markov sources, we weren't counting on polarization here anyway!

The crucial property we identify and exploit in the analysis is the following: the Markovian nature of the source plus the row-by-row arrival ordering of  $Z$ , implies that the distribution of the  $j$ 'th source column  $Z^j$  conditioned on the previous columns  $Z^{<j} = z^{<j}$ , is a *close to a product distribution*, for all but the last few (say  $\varepsilon m$ ) columns.<sup>8</sup>

It turns out that the analysis of the polar transform  $P_m$  only needs independent inputs, which however need not be identically distributed. We are then able to apply the recent analysis from [2], essentially as black box, to argue that  $P_m$  will compress each of the conditioned sources  $Z^j|Z^{<j} = z^{<j}$  to its respective entropy, and also establish fast convergence via quantitatively strong polynomial (in the gap to capacity) upper bounds on the  $m$  needed to achieve this. Further, we automatically benefit from the generality of the analysis in [2], which applies not only to the  $2 \times 2$  transform  $P_2$  at the base case, but in fact any  $k \times k$  transform (satisfying some minimal necessary conditions) over an arbitrary prime field  $\mathbb{F}_q$ . Previous works on polar coding for Markovian sources [4, 9, 12] only applied for Boolean sources.

We remark that the use of the identity transform for the rows in  $\tilde{P}_n$  is quite counterintuitive. It implies that the compression matrix is a block diagonal matrix (after some permutation of the rows and columns) – and in turn this seems to suggest that we are compressing different parts of the input sequence “independently”. However this is not quite true. The relationship between the blocks ends up influencing the final set  $S$  of the bits of  $\tilde{P}_n(Z)$  that are output by the compression algorithm. Furthermore the decompression relies on the information obtained from the decompression of the blocks corresponding to  $Z^{<j}$  to compute the block  $Z^j$ .

**Decompression algorithm.** Our alteration to apply the identity transform for the rows also helps us with the task of decompression. Toward this, we build on a decompression algorithm for memoryless sources from [2] that is somewhat different looking from the usual ones in the polar coding literature. This algorithm aims to compute  $U = P_m^{\text{row}}(Z)$  one *column* at a time, given  $P_n(Z)|_S$ . Given the first  $j - 1$  columns  $U^{<j} = u^{<j}$ , the algorithm first computes the conditional distribution of  $U^j$  conditioned on  $U^{<j} = u^{<j}$  and then uses a recursive decoding algorithm for  $P_m$  to determine  $U^j$ . The key to the recursive use is again that the decoding algorithm works as long as the input variables are independent (and in particular, does not need them to be identically distributed).

In our Markovian setting, we now have to compute the conditional distribution of  $Z^j$  conditioned on  $Z^{<j} = z^{<j}$ . But as mentioned above, this conditional distribution is close to a product distribution, say  $D_j(z^{<j})$  (except for the last few columns  $j$  where decompression is trivial as we output the entire column). Further, the marginals of this product distribution are easily computed using dynamic programming (via what is called the “Forward Algorithm” for hidden Markov models, described for completeness in Appendix A). We can then determine the  $j$ 'th column  $Z^j$  (having already recovered the first  $j - 1$  columns as  $z^{<j}$ ) by running (in a black box fashion) the polar decompressor from [2] for the memoryless case, feeding this product distribution  $D_j(z^{<j})$  as the source distribution.

**Computing the output indices.** Finally we need one more piece to make the result fully constructive. This is the preprocessing needed to compute the subset  $S$  of the coordinates of  $\tilde{P}_n(Z)$  that have noticeable conditional entropy. For the memoryless case these computations

---

<sup>8</sup> We handle the non-independence in the last few columns, by simply outputting those columns  $P_m(Z^j)$  in entirety, rather than only a set  $S_j$  of entropy-carrying positions. This only adds an  $\varepsilon$  fraction to the output length, which we can afford.

**Algorithm 4.1** POLAR-COMPRESS.**Constants:**  $M \in \mathbb{F}_q^{k \times k}$ ,  $m = k^t$ ,  $n = m^2$ **Input:**  $Z = (Z_{11}, Z_{12}, \dots, Z_{mm}) \in \mathbb{F}_q^n$ , and sets  $S_j \subseteq [m]$  for  $j \in [m]$ **Output:**  $U_{S_j}^j \in \mathbb{F}_q^{s_j}$  for all  $j \in [m]$   $\triangleright s_j := |S_j|$  for  $j \leq (1 - \varepsilon)m$ , and  $s_j := m$  otherwise.

---

```

1: procedure POLAR-COMPRESS( $Z; \{S_j\}_{j \in [m]}$ )
2:   for all  $j \in [m]$  do
3:     Compute  $U^j := P_m(Z^j)$ .
4:     If  $j \leq (1 - \varepsilon)m$  then
5:       Output  $U_{S_j}^j$ 
6:     else
7:       Output  $U^j$ 

```

---

were shown to be polynomial time computable in the works of [8, 5, 11]. We manage to extend the ideas from Guruswami and Xia [5] to the case of Markovian channels as well. It turns out the only ingredients needed to make this computation work are, again, the ability to compute the distributions of  $Z^j$  conditioned on  $Z^{<j} = z^{<j}$  for typical values of  $z^{<j}$ . We note that unlike in the setting of memoryless channels (or i.i.d. sources) our preprocessing step is randomized. We believe this is related to the issue that there is no “closed” form solutions to basic questions related to Markovian sources and channels (such as the capacity of the channel in Figure 1) and this forces us to use some random sampling and estimation to compute some of the conditional entropies needed by our algorithms.

**Organization of rest of the paper.** In the next section (Section 4) we describe our compression and decompression algorithms. In Section 5 we describe a notion of “nice”-ness for the preprocessing stage and show that if the preprocessing algorithm returns a nice output, then the compression and decompression algorithm work correctly with moderately high probability (over the message produced by the source). In Section 6 we describe our preprocessing algorithm that returns a nice set with all but exponentially small failure probability (over its internal coin tosses). Finally in Section 7 we give the formal proofs of Theorems 9 and 10.

## 4 Construction

### 4.1 Compression Algorithm

Our compression, decompression and preprocessing algorithms are defined with respect to arbitrary mixing matrices  $M \in \mathbb{F}_q^{k \times k}$ . (Recall that mixing matrices were defined in Definition 8.) Though a reader seeking simplicity may set  $k = 2$  and  $M = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}$ . Given integer  $t$ , let  $m = k^t$  and let  $P_m = P_{m,M} : \mathbb{F}_q^m \rightarrow \mathbb{F}_q^m$  be the polarization transform given by  $P_m = M^{\otimes t}$ .

### 4.2 Fast Decompressor

The decompressor below makes black-box use of the FAST-DECODER from [2, Algorithm 4].

The FAST-DECODER takes as input the description of a *product distribution*  $\mathcal{D}_Z$  on inputs in  $\mathbb{F}_q^m$ , as well as the specified coordinates of the compression  $U$ . It is intended to decode from the encoding  $U' \in \{\mathbb{F}_q \cup \{\perp\}\}^m$ , where  $U := M^{\otimes t}Z$ , coordinates of  $Z$  are independent,

---

**Algorithm 4.2** POLAR-DECOMPRESS.
 

---

**Constants:**  $M \in \mathbb{F}_q^{k \times k}$ ,  $m = k^t$ ,  $n = m^2$ 
**Input:** Markov Source  $\mathcal{H}$  and  $U_{S_1}^1, U_{S_2}^2, \dots, U_{S_m}^m \in \mathbb{F}_q^m$ 
**Output:**  $\hat{Z} \in \mathbb{F}_q^{m \times m}$ 

```

1: procedure POLAR-DECOMPRESS( $\mathcal{H}; U_{S_1}^1, U_{S_2}^2, \dots, U_{S_m}^m$ )
2:   for all  $j \in [m]$  do
3:     If  $j \leq (1 - \varepsilon)m$  then
4:       Compute the distribution  $\mathcal{D}_{z^j | \hat{z}^{<j}} \equiv \bar{Z}^j | \{\bar{Z}^{<j} = \hat{Z}^{<j}\}$ , using the Forward Algorithm on Markov Source  $\mathcal{H}$ .
5:       Define  $U^j \in \{\mathbb{F}_q \cup \{\perp\}\}^m$  by extending  $U_{S_j}^j$  using  $\perp$  in the unspecified coordinates.
6:       Set  $\hat{Z}^j \leftarrow \text{FAST-DECODER}(\mathcal{D}_{z^j | \hat{z}^{<j}}; U^j)$ 
7:     else
8:       Set  $\hat{Z}^j \leftarrow (M^{-1})^{\otimes t} \hat{U}_{S_j}^j$  ▷ Note here  $S_j = [m]$ 
9:   Return  $\hat{Z}$ 

```

---

and  $U'$  is defined by  $U$  on the high-entropy coordinates of  $U$  (and  $\perp$  otherwise). It outputs an estimate  $\hat{Z}$  of the input  $Z$ .

Note that, for a Markov source  $\mathcal{H}$  on  $\ell$  states, Line 4 takes time  $\mathcal{O}_q(m^2 \ell^2)$  (time  $\mathcal{O}_q(m \ell^2)$  per coordinate of  $\bar{Z}^j$ , using the Forward Algorithm). The FAST-DECODER call in Line 6 takes time  $\mathcal{O}_q(m \log m)$ . Thus, the total runtime is  $\mathcal{O}_q(m^3 \ell^2 + m^2 \log m) = \mathcal{O}_q(n^{3/2} \ell^2 + n \log n)$ .

## 5 Analysis

The goal of this section is to prove that the decompressor works correctly, with high probability, provided the preprocessing stage returns the appropriate sets  $\{S_j\}$ . Specifically, we prove Theorem 12 as stated below. But first we need a definition of “nice” sets  $\{S_j\}$ : We will later show that pre-processing produces such sets and compression and decompression work correctly (w.h.p.) given nice sets.

► **Definition 11** ( $(\varepsilon, \zeta)$ -niceness). Let  $\mathcal{H}$  be a Markov source. For every  $m \in \mathbb{N}$  and  $n = m^2$ , let  $\bar{Z} \sim \mathcal{H}_m^{\otimes m}$  be the corresponding “independent” distribution. Let  $\bar{U} := P_m^{\text{column}}(\bar{Z})$ .

We call sets  $S_1, S_2, \dots, S_m \subseteq [m]$  “ $(\varepsilon, \zeta)$ -nice” if they satisfy the following:

1.  $\sum_j |S_j| \leq \bar{H}(Z) + \varepsilon n$
2.  $\forall j \in [m], i \notin S_j : \bar{H}(\bar{U}_{(i,j)} | \bar{U}_{\prec(i,j)}) < \zeta$

Now, the rest of this section will show the following.

► **Theorem 12.** *There exists a polynomial  $p(\cdot)$  such that for every  $\varepsilon > 0$ ,  $\tau > 0$ , and  $n = m^2 > p(\tau/\varepsilon)$  the following holds:*

*Let  $\mathcal{H}$  be an aperiodic irreducible Markov source with alphabet  $\mathbb{F}_q$ , mixing time  $\tau$  and underlying state space  $[\ell]$ . Define random variables  $Z = (Z_{11}, Z_{12}, \dots, Z_{mm}) \sim \mathcal{H}_{m^2}$  as generated by  $\mathcal{H}$ . Then, for all sets  $S_1, S_2, \dots, S_m \subseteq [m]$  that are  $(\varepsilon, \zeta)$ -nice as per Definition 11, we have:*

$$\Pr_Z[\text{POLAR-DECOMPRESS}(\text{POLAR-COMPRESS}(Z; \{S_j\}_{j \in [m]})) \neq Z] \leq n\zeta + m \exp(-\varepsilon m/\tau)$$

## 5.1 Proof Overview

Throughout this section, let  $\mathcal{H}$  be a stationary Markov source with alphabet  $\mathbb{F}_q$  and mixing-time  $\tau$ . The key part of the analysis is showing that compression and decompression succeed when applied to the “independent” distribution  $\bar{Z} \sim \mathcal{H}_m^{\otimes m}$ . To do this, we first show that the compression transform “polarizes” entropies, which follows directly from the results of [2, 3]. Then we show that, provided “nice” sets can be computed (low-entropy sets, a la Definition 11), the compression and decompression succeed with high probability. This also follows essentially in a black-box fashion from the results of [2]. Finally, we argue that the compression and decompression also work for the actual distribution  $Z \sim \mathcal{H}_{m^2}$ , simply by observing that the involved variables are close in distribution.

We later describe how such “nice” sets can be computed in polynomial time, given the description of the Markov source  $\mathcal{H}$ .

## 5.2 Polarization

In this section, we show that the compression transform  $P_m^{\text{column}}$  polarizes entropies.

► **Lemma 13.** *Let  $\mathcal{H}$  be a Markov source, and let  $\bar{Z} \sim \mathcal{H}_m^{\otimes m}$ . Let  $\bar{U} = P_m^{\text{column}}(\bar{Z})$ .*

*Then, there exists a polynomial  $p(\cdot)$  such that for every  $\varepsilon > 0$ , there exists  $\beta > 0$  such that if  $m > p(1/\varepsilon)$ , the following holds: For all but  $\varepsilon$ -fraction of indices  $i, j \in [m] \times [m]$ , the normalized entropy*

$$\bar{H}(\bar{U}_{i,j} | \bar{U}_{\prec(i,j)}) \notin (\exp(-m^\beta), 1 - \varepsilon)$$

**Proof.** We will show that for each column  $\bar{U}^j$ , all but  $\varepsilon$ -fraction of indices  $i \in [m]$  have entropies

$$\bar{H}(\bar{U}_i^j | \bar{U}_{\prec i}^j, \bar{U}^{\prec j}) \notin (\exp(-m^\beta), 1 - \varepsilon)$$

Indeed, this follows directly from the analysis in [3]. For each  $j$ , the set of variables  $(\bar{Z}_1^j, \bar{Z}_1^{\prec j}), (\bar{Z}_2^j, \bar{Z}_2^{\prec j}), \dots, (\bar{Z}_m^j, \bar{Z}_m^{\prec j})$  are independent and identically distributed. Thus, Theorem 14 from [3] (reproduced below) implies that the conditional entropies are polarized. Specifically, let  $p(\cdot)$  and  $\beta$  be as guaranteed by Theorem 14, for the distribution  $\mathcal{D} \equiv (\bar{Z}_1^j, \bar{Z}_1^{\prec j})$ . Then, since  $P_m = M^{\otimes t}$ , we have

$$\begin{aligned} \bar{H}(\bar{U}_i^j | \bar{U}_{\prec i}^j, \bar{U}^{\prec j}) &= \bar{H}(\bar{U}_i^j | \bar{U}_{\prec i}^j, P_m(\bar{Z}^{\prec j})) && \text{(by definition)} \\ &= \bar{H}(\bar{U}_i^j | \bar{U}_{\prec i}^j, \bar{Z}^{\prec j}) && (P_m \text{ is invertible}) \\ &\notin (\exp(-m^\beta), 1 - \varepsilon) && \text{(Theorem 14)} \end{aligned}$$

◀

The following theorem is direct from the works [3].

► **Theorem 14.** *For every  $k \in \mathbb{N}$ , prime  $q$ , mixing-matrix  $M \in \mathbb{F}_q^{k \times k}$ , discrete set  $\mathcal{Y}$ , and any distribution  $\mathcal{D} \in \Delta(\mathbb{F}_q \times \mathcal{Y})$ , the following holds. Define the random vectors  $A := (A_1, A_2, \dots, A_n)$  and  $B := (B_1, B_2, \dots, B_n)$  where  $n = k^t$  and each component  $(A_i, B_i)$  is independent and identically distributed  $(A_i, B_i) \sim \mathcal{D}$ .*

*Let  $X := M^{\otimes t} A$ . Then, the conditional entropies of  $X$  are polarized: There exists a polynomial  $p(\cdot)$  and  $\beta > 0$  such that for every  $\varepsilon > 0$ , if  $n = k^t > p(1/\varepsilon)$ , then all but  $\varepsilon$ -fraction of indices  $i \in [n]$  have normalized entropy*

$$\bar{H}(X_i | X_{\prec i}, B) \notin (\exp(-n^\beta), 1 - \varepsilon).$$



### 5.3 Independent Analysis

Now we bound the failure probability of the Polar Compressor and Decompressor, when applied to the “independent” input distribution  $\bar{Z}$ .

► **Claim 15.** *Let  $\bar{Z} \sim \mathcal{H}_m^{\otimes m}$  and  $\bar{U} := P_m^{\text{column}}(\bar{Z})$ . Then, for all sets  $S_1, S_2, \dots, S_m \subseteq [m]$ ,*

$$\Pr_{\bar{U} \leftarrow P_m^{\text{column}}(\bar{Z})} [\text{POLAR-DECOMPRESS}(U_{S_1}^1, U_{S_2}^2, \dots, U_{S_m}^m) \neq \bar{U}] \leq \sum_{j \in [m], i \notin S_j} \bar{H}(\bar{U}_i^j | \bar{U}_{<i}^j, \bar{U}^{<j})$$

**Proof.** Appears in the full version of this paper. ◀

### 5.4 Proof of Main Theorem

At this point, we can show the entire process of compression and decompression succeeds with high probability, proving Theorem 12.

First, we argue  $\mathcal{H}_{m^2}$  and  $\mathcal{H}_m^{\otimes m}$  are close in the appropriate sense.

► **Lemma 16.** *Let  $Z \sim \mathcal{H}_{m^2}$  and  $\bar{Z} \sim \mathcal{H}_m^{\otimes m}$ . Then, for every  $\ell \in [m]$ , the distribution of  $Z^{<m-\ell}$  and  $\bar{Z}^{<m-\ell}$  are  $m \cdot \exp(-\ell/\tau)$ -close in  $L_1$ .*

**Proof.** We proceed by a sequence of  $m$  hybrids, changing one row at a time to being independent. Let the  $i$ -th hybrid be  $H_i := Z_{\leq i}^{<m-\ell} \circ \bar{Z}_{>i}^{<m-\ell}$ , that is, the first  $i$  rows of  $Z$ , with the remaining rows replaced by iid copies of  $Z_1$ .

Consider moving from  $H_{i+1}$  to  $H_i$ . Conditioned on the first  $i$  rows of  $Z^{<m-\ell}$ , the distribution of the hidden state of the Markov source, at the beginning of the  $(i+1)$ th row, is  $\exp(-\ell/\tau)$ -close to its stationary distribution  $\pi$  (since  $\ell$  steps pass between  $Z_{i,m-\ell}$  and  $Z_{i+1,1}$ ). Recall that the distribution of  $Z_1$  is generated by the Markov source starting from  $\pi$ . Thus, the distribution of the  $(i+1)$ th row of  $Z$ , conditioned on the first  $i$  rows of  $Z^{<m-\ell}$ , is  $\exp(-\ell/\tau)$ -close to the distribution of  $Z_1$ . So,  $|H_{i+1} - H_i|_1 \leq \exp(-\ell/\tau)$ . Since we pass through  $m$  hybrids, the total  $L_1$  distance is at most  $m \cdot \exp(-\ell/\tau)$ . ◀

**Proof of Theorem 12.** First, we show the corresponding claim about the “independent” distribution  $\bar{Z} \sim \mathcal{H}_m^{\otimes m}$ :

► **Claim 17.** *For  $\bar{Z} \sim \mathcal{H}_m^{\otimes m}$ , we have:*

$$\Pr_{\bar{U} \leftarrow P_m^{\text{column}}(\bar{Z})} [\text{POLAR-DECOMPRESS}(\bar{U}_{S_1}^1, \bar{U}_{S_2}^1, \dots, \bar{U}_{S_m}^m) \neq \bar{Z}] \leq n\zeta$$

or equivalently,

$$\Pr_{\bar{Z} \sim \mathcal{H}_m^{\otimes m}} [\text{POLAR-DECOMPRESS}(\text{POLAR-COMPRESS}(\bar{Z}; \{S_j\}_{j \in [m]})) \neq \bar{Z}] \leq n\zeta$$

**Proof.** By Claim 15, we have

$$\begin{aligned} \Pr_{\bar{U} \leftarrow P_m^{\text{column}}(\bar{Z})} [\text{POLAR-DECOMPRESS}(\bar{U}_{S_1}^1, \bar{U}_{S_2}^1, \dots, \bar{U}_{S_m}^m) \neq \bar{Z}] &\leq \sum_{j \in [m], i \notin S_j} \bar{H}(\bar{U}_i^j | \bar{U}_{<i}^j, \bar{U}^{<j}) \\ &\leq n\zeta \quad (\text{by } (\varepsilon, \zeta)\text{-niceness}) \end{aligned} \quad \blacktriangleleft$$

Continuing the proof of Theorem 12, notice that the composition of POLAR-COMPRESS and POLAR-DECOMPRESS always operate as the identity transform on the inputs  $Z^j$  for  $j > (1-\varepsilon)m$ . Thus, it suffices to consider the behavior of this composition on inputs  $Z^{\leq (1-\varepsilon)m}$ . In this case, Lemma 16 guarantees that the distributions of  $\bar{Z}^{\leq (1-\varepsilon)m}$  and  $Z^{\leq (1-\varepsilon)m}$  are close in  $L_1$ , and thus we may conclude by Claim 17:

$$\begin{aligned}
& \Pr_{Z \sim \mathcal{H}_{m^2}} [\text{POLAR-DECOMPRESS}(\text{POLAR-COMPRESS}(Z; \{S_j\}_{j \in [m]})) \neq Z] \\
& \leq \Pr_{\bar{Z} \sim \mathcal{H}_m^{\otimes m}} [\text{POLAR-DECOMPRESS}(\text{POLAR-COMPRESS}(\bar{Z}; \{S_j\}_{j \in [m]})) \neq \bar{Z}] + m \exp(-\varepsilon m / \tau) \\
& \leq n\zeta + m \exp(-\varepsilon m / \tau) \quad (\text{Claim 17}) \quad \blacktriangleleft
\end{aligned}$$

## 6 Preprocessing

In this section, we describe a pre-processing algorithm to find the  $(\varepsilon, \zeta)$ -nice sets, as defined in Definition 11, that are required by the compression and decompression algorithms. Recall the notion of a mixing matrix (Definition 8). The following theorem shows that for every prime alphabet  $\mathbb{F}_q$  and mixing matrix  $M \in \mathbb{F}_q^{k \times k}$ , there is an efficient algorithm that can find nice sets in polynomial time. Specifically, we prove the following theorem.

► **Theorem 18.** *For every prime  $q$  and mixing-matrix  $M \in \mathbb{F}_q^{k \times k}$ , there exists a polynomial  $p(\cdot)$  and a polynomial time preprocessing algorithm POLAR-PREPROCESS (Algorithm 6.3), such that for every  $\varepsilon > 0$  and  $m > p(1/\varepsilon)$ , the following holds:*

*Let  $\mathcal{H}$  be a Markov source with mixing-time  $\tau$ , alphabet  $\mathbb{F}_q$ , and underlying state space  $[\ell]$ . Let  $\bar{Z} \sim \mathcal{H}_m^{\otimes m}$  for  $m = k^t$ , and  $\bar{U} := P_m^{\text{column}}(\bar{Z})$ .*

*Let*

$$S_1, S_2, \dots, S_m \leftarrow \text{POLAR-PREPROCESS}(q, k, t, M, \mathcal{H})$$

*Then, except with probability  $\exp(-\Omega(m))$  over the randomness of the algorithm, the output sets  $S_1, S_2, \dots, S_m \subseteq [m]$  are  $(\varepsilon, \zeta = \mathcal{O}(\frac{1}{n^3}))$ -nice for  $\mathcal{H}$ . Further, the algorithm runs in time  $\text{poly}_q(m, \ell, 1/\varepsilon)$ .*

Our main goal will be to estimate the conditional entropies

$$\bar{H}(\bar{U}_{(i,j)} | \bar{U}_{\prec(i,j)}) = \bar{H}(\bar{U}_{i,j} | \bar{U}_{\prec i,j}, \bar{U}^{\prec j}) = \bar{H}(\bar{U}_{i,j} | \bar{U}_{\prec i,j}, \bar{Z}^{\prec j})$$

for  $\bar{Z} \sim \mathcal{H}_m^{\otimes m}$  and  $\bar{U} := P_m^{\text{column}}(\bar{Z})$ . Then, we will construct the “nice” sets by defining, for each  $j$ ,  $S_j$  as the set of indices with high entropy:  $S_j := \{i \in [m] : \bar{H}(\bar{U}_{i,j} | \bar{U}_{\prec i,j}, \bar{Z}^{\prec j}) > \frac{1}{n^3}\}$ . By Polarization (Lemma 13), these sets have size at most  $\sum_j |S_j| \leq \bar{H}(Z) + \varepsilon n$ , since they must have conditional entropies close to 1 (except possibly for some  $\varepsilon$  fraction of indices  $(i, j) \in [m] \times [m]$ ).

We will estimate conditional entropies  $\bar{H}(\bar{U}_{i,j} | \bar{U}_{\prec i,j}, \bar{Z}^{\prec j})$  by approximately tracking the distribution of variables as we apply successive tensor-powers of  $M$ . Since we are only interested in conditional entropies, it is sufficient to “quantize” the true distribution of, for example  $U_i | U_{\prec i}$ , into an approximation  $U_i | A$ , such that  $H(U_i | U_{\prec i}) \approx H(U_i | A)$ . This algorithm follows the same high-level strategy of [5], of approximating the conditional distributions via quantized bins. It turns out that this strategy can be implemented for Markov sources, using the fact that Markov sources are constructive. We define our notions of approximation, and formalize this strategy below.

### 6.1 Notation and Preliminaries

► **Definition 19** (Associated Conditional Distribution). Let  $X$  be a random variable taking values in universe  $U$ , and let  $W$  be an arbitrary random variable. Let  $\mathcal{D}_{X|w} \in \Delta(U)$  denote the conditional distribution of  $X | \{W = w\}$ . Let  $\mathbb{D}_{X|W} \in \Delta(\Delta(U))$  be the distribution over  $\mathcal{D}_{X|w}$  defined by sampling  $w \sim W$ . We call  $\mathbb{D}_{X|W}$  the *associated conditional distribution to  $X|W$* .

As above, we use boldface  $\mathbb{D}$  to denote objects of type  $\Delta(\Delta(U))$ . Note that we can operate on conditional distributions as we would on their underlying random variables. For example, for random variables  $(A_1, W)$  and  $(A_2, Y)$  such that  $A_1, A_2 \in \mathbb{F}_q$  and  $(A_1, W)$  is independent from  $(A_2, Y)$ , the associated conditional distribution of  $A_1 + A_2|Y, W$  can be computed from the associated conditional distributions of  $A_1|Y$  and  $A_2|W$ . To more easily describe such operations on conditional distributions (which may not always arise from underlying random variables), we define the *implicit random variables* associated to a conditional distribution:

► **Definition 20** (Implicit Random Variables Associated to Conditional Distribution). For every  $\mathbb{D}_{X|W} \in \Delta(\Delta(U))$ , define *implicit random variables*  $X, W$  associated to  $\mathbb{D}_{X|W}$  as random variables  $(X, W)$  such that the associated conditional distribution to  $X|W$  is exactly  $\mathbb{D}_{X|W}$ . Note that there is not a unique choice of such random variables.

Using this, we can naturally define (for example)  $\mathbb{D}_{A_1+A_2|W, Y}$  and  $\mathbb{D}_{A_2|W, Y, A_1+A_2}$  from any  $\mathbb{D}_{A_1, W}, \mathbb{D}_{A_2, Y} \in \Delta(\Delta(\mathbb{F}_q))$ . Note that we will always be performing such operations assuming independence of the involved implicit random variables, ie  $(A_1, W)$  and  $(A_2, Y)$ .

► **Definition 21** (Conditional Distance). Let  $(X, W)$  and  $(Y, Z)$  be two joint distributions, such that  $X$  and  $Y$  take values in the same universe  $U$ . Let  $\mathbb{D}_{X|W}$  and  $\mathbb{D}_{Y|Z}$  be the associated distributions in  $\Delta(\Delta(U))$ . Then, define the *conditional distance*

$$d_C(\mathbb{D}_{X|W}, \mathbb{D}_{Y|Z}) := \min_{\substack{(A, B): \text{a distribution in } \Delta(\Delta(U) \times \Delta(U)) \\ \text{s.t. marginals of } A \text{ match } \mathbb{D}_{X|W}, \text{ and} \\ \text{marginals of } B \text{ match } \mathbb{D}_{Y|Z}}} \mathbb{E}_{(D_A, D_B) \sim (A, B)} [\|D_A - D_B\|_1]$$

Note that  $d_C$  can be equivalently defined as an optimal transportation cost between two distributions in  $\Delta(\Delta(U))$ , where the cost of moving a unit of mass between points  $D_i, D_j \in \Delta(U)$  is  $\|D_i - D_j\|_1$ .

This metric behaves naturally under post-processing:

► **Claim 22.** For all  $\mathbb{D}_{X|W}, \mathbb{D}_{X'|W'} \in \Delta(\Delta(U))$ , and any  $f: U \rightarrow V$ ,

$$d_C(\mathbb{D}_{f(X)|W}, \mathbb{D}_{f(X')|W'}) \leq d_C(\mathbb{D}_{X|W}, \mathbb{D}_{X'|W'})$$

For computational purposes, we represent the space of distributions using  $\varepsilon$ -nets:

► **Definition 23** ( $\varepsilon$ -nets). For every set  $U$  and any  $\varepsilon > 0$ , let  $T_\varepsilon(U) \subseteq \Delta(U)$  be an  $\varepsilon$ -net of  $\Delta(U)$  with respect to  $L_1$ . That is, for every  $\mathcal{D} \in \Delta(U)$ , there exists  $\hat{\mathcal{D}} \in T_\varepsilon(U)$  such that  $\|\mathcal{D} - \hat{\mathcal{D}}\|_1 \leq \varepsilon$ .

Note that for  $|U| = |\mathbb{F}_q| = q$ ,  $T_\varepsilon(U)$  can be chosen such that  $|T_\varepsilon(U)| \leq \left(\frac{q}{\varepsilon} + q\right) \leq \left(\frac{2q}{\varepsilon}\right)^q = \text{poly}_q(1/\varepsilon)$ .

Moreover,  $\Delta(T_\varepsilon(U))$  is an  $\varepsilon$ -net of  $\Delta(\Delta(U))$  under the  $d_C$ -metric.

## 6.2 Conditional Distribution Approximation

The below procedure takes as input a conditional distribution  $\mathbb{D}_{Z|W} \in \Delta(\Delta(\mathbb{F}_q))$ , and computes an approximation to the conditional distribution of  $U_I|(U_{\setminus I}, W_1, \dots, W_{k^t})$ , for an index  $I \in [k]^t$ , where  $U := M^{\otimes t}Z$  and  $\{(Z_i, W_i)\}_{i \in [k^t]}$  are independently defined by  $\mathbb{D}_{Z|W}$ .

Note that if the input  $\mathbb{D}_{Z|W}$  is specified in an  $\varepsilon$ -net  $\Delta(T_\varepsilon(\mathbb{F}_q))$ , then the above procedure runs in time  $\text{poly}_q(m, 1/\varepsilon)$  for  $m = k^t$ .

**Algorithm 6.1** Conditional Distribution Approximation.

---

**Input:** Conditional distribution on inputs  $\mathbb{D}_{Z|W} \in \Delta(\Delta(\mathbb{F}_q))$ ,  $\varepsilon > 0$ ,  $t \in \mathbb{N}$ , index  $I \in [k]^t$ , and  $M \in \mathbb{F}_q^{k \times k}$

**Output:** Conditional distribution  $\tilde{\mathbb{D}}_{U|W} \in \Delta(\Delta(\mathbb{F}_q))$ , an approximation to  $U_I | (U_{\prec I}, W_1, \dots, W_{k^t})$  for  $U := M^{\otimes t} Z$  and  $\{(Z_i, W_i)\}_{i \in [k^t]}$  independently defined by  $\mathbb{D}_{Z|W}$ .

- 1: **procedure** APPROXDIST( $\mathbb{D}_{Z|W}, \varepsilon, t, I = (I_1, \dots, I_t), M$ )
- 2:   **If**  $t = 0$  **then**
- 3:     **Return**  $\mathbb{D}_{Z|W}$
- 4:   **else**
- 5:      $\hat{\mathbb{D}}_{Z|Y} \leftarrow$  APPROXDIST( $\mathbb{D}_{Z|W}, \varepsilon/(2k), t-1, I_{\prec t} = (I_1, \dots, I_{t-1}), M$ )
- 6:      $j \leftarrow I_t$ .
- 7:     Explicitly compute the following conditional distribution  $\hat{\mathbb{D}}_{U_j | U_{\prec j}, Y_1, \dots, Y_k} \in \Delta(\Delta(\mathbb{F}_q))$ :
- 8:       Let  $(Z, Y)$  be the implicit random variables associated to  $\hat{\mathbb{D}}_{Z|Y}$ .
- 9:       Let  $\{(Z_i, Y_i)\}_{i \in [k]}$  be independent random variables distributed identically to  $(Z, Y)$ .
- 10:       Define random vector  $U := M \cdot Z'$ , Where  $Z' = (Z_1, \dots, Z_k)$ .
- 11:       Let  $\hat{\mathbb{D}}_{U_j | U_{\prec j}, Y_1, \dots, Y_k}$  be the associated conditional distribution to  $U_j | U_{\prec j}, Y_1, \dots, Y_k$ .
- 12:       Round  $\hat{\mathbb{D}}_{U_j | U_{\prec j}, Y_1, \dots, Y_k}$  to  $\tilde{\mathbb{D}}_{U|Y} \in \Delta(T_{\varepsilon/2}(\mathbb{F}_q))$ , a point in the  $\varepsilon/2$ -net of  $\Delta(\Delta(\mathbb{F}_q))$  under  $d_C$ .
- 13:     **Return**  $\tilde{\mathbb{D}}_{U|Y}$ .

---

► **Lemma 24.** For all  $\mathbb{D}_{Z|W} \in \Delta(\Delta(U))$ ,  $\varepsilon > 0$ ,  $t \in \mathbb{N}$ ,  $M \in \mathbb{F}_q^{k \times k}$ , and  $I \in [k]^t$ , we have

$$d_C(\text{APPROXDIST}(\mathbb{D}_{Z|W}, \varepsilon, t, I, M), \mathbb{D}_{U_I | U_{\prec I}, W_1, \dots, W_{k^t}}) \leq \varepsilon$$

where  $\mathbb{D}_{U_I | U_{\prec I}, W_1, \dots, W_{k^t}}$  is the associated conditional distribution to the random variables defined as follows. Let  $(Z, W)$  be the implicit random variables associated to  $\mathbb{D}_{Z|W}$ . Let  $\{(Z_i, W_i)\}_{i \in [k^t]}$  be independent random variables distributed identically to  $(Z, W)$ . Finally, define random vector  $U := M^{\otimes t} \cdot Z'$ , where  $Z' = (Z_1, \dots, Z_{k^t})$ .

**Proof.** Appears in the full version of this paper. ◀

### 6.3 Approximating Conditional Entropies

Here we use Algorithm 6.1 directly to approximate conditional entropies:

► **Theorem 25.** For every field  $\mathbb{F}_q$ , conditional distribution  $\mathbb{D}_{Z|W} \in \Delta(\Delta(\mathbb{F}_q))$ , matrix  $M \in \mathbb{F}_q^{k \times k}$ ,  $t \in \mathbb{N}$ ,  $m = k^t$ , and  $\gamma > 0$ , consider the random variable  $U := M^{\otimes t} Z$  where each  $\{(Z_i, W_i)\}_{i \in [m]}$  is sampled independently from  $\mathcal{D}_{Z,W}$ .

Then, Algorithm 6.2 outputs  $\hat{h}_1, \dots, \hat{h}_m \leftarrow$  APPROXENTROPY( $\mathbb{D}_{Z|W}, \gamma, t, M$ ) such that

$$\forall i \in [m] : \hat{h}_i = \overline{H}(U_i | U_{\prec i}, W_1, \dots, W_m) \pm \gamma$$

Further, if the input  $\mathbb{D}_{Z|W}$  is specified in an  $\varepsilon$ -net  $\Delta(T_\varepsilon(\mathbb{F}_q))$ , then the above procedure runs in time  $\text{poly}_q(m, 1/\varepsilon, 1/\gamma)$ .

**Algorithm 6.2** Entropy Approximation.

---

**Input:**  $\gamma > 0$ ,  $t \in \mathbb{N}$ , Conditional distribution  $\mathbb{D}_{Z|W} \in \Delta(\Delta(\mathbb{F}_q))$ , and  $M \in \mathbb{F}_q^{k \times k}$

**Output:**  $\{\hat{h}_i \in \mathbb{R}\}_{i \in [k^t]}$

- 1: **procedure** APPROXENTROPY( $\mathbb{D}_{Z|W}, \gamma, t, M$ )
- 2:    $m \leftarrow k^t$
- 3:    $\varepsilon \leftarrow \frac{\gamma^2}{16 \log(q)}$
- 4:   **for all**  $I \in [k]^t$  **do**
- 5:      $\mathbb{D}_{U|Y} \leftarrow \text{APPROXDIST}(\mathbb{D}_{Z|W}, \varepsilon, t, I, M)$
- 6:      $\hat{h}_I \leftarrow \overline{H}(U|Y)$ , the conditional entropy of the implicit random variables  $(U, Y)$  associated to  $\mathbb{D}_{U|Y}$ .
- 7:   **Return**  $\{\hat{h}_i\}_{i \in [k^t]}$                     $\triangleright$  Abusing notation by identifying  $[k]^t$  with  $[k^t]$ .

---

**Algorithm 6.3** POLAR-PREPROCESS.

---

**Input:**  $q, k, t \in \mathbb{N}$  with  $q$  prime,  $M \in \mathbb{F}_q^{k \times k}$ , and Markov source  $\mathcal{H}$                     $\triangleright m = k^t, n = m^2$

**Output:** Sets  $S_1, S_2, \dots, S_m \subseteq [m]$

- 1: **procedure** POLAR-PREPROCESS( $q, k, t, M, \mathcal{H}$ )
- 2:    $m \leftarrow k^t$ ;  $\gamma \leftarrow \frac{1}{n^{10}}$ ;  $N \leftarrow |T_\gamma(\mathbb{F}_q)|$ ;  $R \leftarrow n(N/\gamma)^2$                     $\triangleright N \leq \text{poly}_q(1/\gamma)$
- 3:   **for all**  $j \in [m]$  **do**
- 4:     **for all**  $i = 1, 2, \dots, R$  **do**
- 5:       Sample a sequence  $w_i := (y_1, y_2, \dots, y_{j-1})$  from  $\mathcal{H}$ .
- 6:       Compute  $\mathcal{D}_{w_i} \in \Delta(\mathbb{F}_q)$ , the distribution of  $Y_j | Y_{<j} = w_i$ , using the Forward Algorithm A.1 for  $\mathcal{H}$ .
- 7:       Let  $\tilde{\mathbb{D}}_{Y|W} \in \Delta(\Delta(\mathbb{F}_q))$  be the empirical distribution of  $\mathcal{D}_w$ , from the samples  $\mathcal{D}_{w_i}$  above.
- 8:        $\{\hat{h}_1, \dots, \hat{h}_m\} \leftarrow \text{APPROXENTROPY}(\tilde{\mathbb{D}}_{Y|W}, \gamma = \frac{1}{n^4}, t, M)$
- 9:        $S_j \leftarrow \{i \in [m] : \hat{h}_i > \frac{1}{n^3}\}$
- 10: **Return**  $S_1, S_2, \dots, S_j$ .

---

**Proof of Theorem 25.** Correctness of Algorithm 6.2 follows from the fact that  $O(\frac{\gamma^2}{\log(q)})$ -closeness in the  $d_C$ -metric implies  $\gamma$ -closeness of conditional entropies. Thus, using Algorithm 6.1 to approximate the conditional distributions within  $O(\frac{\gamma^2}{\log(q)})$  is sufficient. See the full version of this paper for details.  $\blacktriangleleft$

## 6.4 Nice Subset Selection

Now we can describe how to find “nice” sets. We first approximate the conditional distribution  $\mathbb{D}_{Z_t|Z_{<t}} \in \Delta(\Delta(\mathbb{F}_q))$  for  $Z_1, \dots, Z_t \sim \mathcal{H}_t$ , by sampling. This crucially relies on the fact that  $\mathcal{H}$  is a constructive source (ie, using the Forward Algorithm). Then we use Algorithm 6.2 to estimate conditional entropies, and select high-entropy indices.

The correctness of this algorithm (proof of Theorem 18) appears in the full version of this paper.

## 7 Proofs of Theorems 9 and 10

Combining Theorem 18 (to compute nice sets) with Theorem 12 (compressing and decompressing assuming nice sets), Theorem 9 follows immediately.

**Proof of Theorem 9.** The algorithms claimed are Algorithm 6.3 for preprocessing, Algorithm 4.1 for compressing and Algorithm 4.2 for decompression. Theorem 18 asserts that Algorithm 6.3 returns a nice sequence of sets  $S_1, \dots, S_m$  with all but exponentially small probability in  $n$ . And Theorem 12 asserts that if  $S_1, \dots, S_m$  are nice then Algorithm 4.1 and 4.2 compress and decompress correctly with high probability over the output of the Markovian source. This yields the theorem. ◀

Finally we show how Theorem 10 follows from Theorem 9.

**Proof of Theorem 10.** Let  $H \in \mathbb{F}_q^{s \times n}$  be the matrix specifying the (linear) compression scheme given by the Preprocessing Algorithm in Theorem 9, when applied to Markov source  $\mathcal{H}$ . The code  $C$  for the additive Markov Channel  $\mathcal{C}_{\mathcal{H}}$  is simply specified by the nullspace of  $H$ , ie encoding is given by  $C(x) := Nx$  where  $N \in \mathbb{F}_q^{n \times n-s}$  spans  $\text{Null}(A)$ .

Note that due to the structure of  $H$ , a nullspace matrix  $N$  can be applied in  $\mathcal{O}_q(n \log n)$  time. In particular,  $H$  is a subset of rows of the block-diagonal matrix  $P \in \mathbb{F}_q^{n \times n}$ , where each  $\sqrt{n} \times \sqrt{n}$  block is the tensor-power  $M^{\otimes t}$ . Thus,  $P^{-1}$  is also block-diagonal with blocks  $(M^{-1})^{\otimes t}$ , and so can be applied in time  $\mathcal{O}_q(n \log n)$ . The matrix  $N$  can be chosen as just a subset of columns of  $P^{-1}$ , and hence can also be applied in time  $\mathcal{O}_q(n \log n)$ .

Let  $y_1, y_2, \dots, y_n \in \mathbb{F}_q$  be distributed according to  $\mathcal{H}$ , and  $y := (y_1, \dots, y_n) \in \mathbb{F}_q^n$ . To decode from  $z = Nx + y$ , the decoder first applies  $H$  (by running the compression algorithm of Theorem 9), to compute  $H z = H N x + H y = H y$ . Then, the decoder runs the decompression algorithm of Theorem 9 on  $H y$  to determine  $y$ . Finally, the decoder can compute  $y - z$  to find the codeword sent ( $Nx$ ), and thus determine  $x$ . (Again using the structure of  $P$ , as above, to determine  $x$  from  $Nx$  in  $\mathcal{O}_q(n \log n)$  time). ◀

---

### References

- 1 Erdal Arıkan. Channel Polarization: A method for constructing capacity-achieving codes for symmetric binary-input memoryless channels. *IEEE Transactions on Information Theory*, pages 3051–3073, July 2009.
- 2 Jarosław Błasiok, Venkatesan Guruswami, Preetum Nakkiran, Atri Rudra, and Madhu Sudan. General strong polarization. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing*, pages 485–492. ACM, 2018. [arXiv:1802.02718](https://arxiv.org/abs/1802.02718).
- 3 Jaroslaw Blasiok, Venkatesan Guruswami, and Madhu Sudan. Polar Codes with Exponentially Small Error at Finite Block Length. In *LIPICs-Leibniz International Proceedings in Informatics*, volume 116. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2018.
- 4 Eren Şaşıoğlu. *Polar coding theorems for discrete systems*. PhD thesis, Ecole Polytechnique Fédérale de Lausanne, 2011.
- 5 Venkatesan Guruswami and Patrick Xia. Polar Codes: Speed of Polarization and Polynomial Gap to Capacity. *IEEE Trans. Information Theory*, 61(1):3–16, 2015. Preliminary version in Proc. of FOCS 2013.
- 6 Seyed Hamed Hassani, Kasra Alishahi, and Rüdiger L. Urbanke. Finite-Length Scaling for Polar Codes. *IEEE Trans. Information Theory*, 60(10):5875–5898, 2014. doi:10.1109/TIT.2014.2341919.

- 7 Satish Babu Korada, Eren Sasoglu, and Rüdiger L. Urbanke. Polar Codes: Characterization of Exponent, Bounds, and Constructions. *IEEE Transactions on Information Theory*, 56(12):6253–6264, 2010. doi:10.1109/TIT.2010.2080990.
- 8 Ramtin Pedarsani, Seyed Hamed Hassani, Ido Tal, and Emre Telatar. On the construction of polar codes. In *Proceedings of 2011 IEEE International Symposium on Information Theory*, pages 11–15, 2011. doi:10.1109/ISIT.2011.6033724.
- 9 Eren Sasoglu and Ido Tal. Polar coding for processes with memory. In *Proceedings of the IEEE International Symposium on Information Theory*, pages 225–229, 2016.
- 10 Boaz Shuval and Ido Tal. Fast Polarization for Processes with Memory. In *Proceedings of the IEEE International Symposium on Information Theory*, pages 851–855, 2018.
- 11 Ido Tal and Alexander Vardy. How to Construct Polar Codes. *IEEE Transactions on Information Theory*, 59(10):6562–6582, October 2013.
- 12 Runxin Wang, Junya Honda, Hirosuke Yamamoto, Rongke Liu, and Yi Hou. Construction of polar codes for channels with memory. In *Proceedings of the 2015 IEEE Information Theory Workshop - Fall (ITW)*, pages 187–191, 2015.
- 13 Runxin Wang, Rongke Liu, and Yi Hou. Joint Successive Cancellation Decoding of Polar Codes over Intersymbol Interference Channels. *CoRR*, 2014. arXiv:1404.3001.

## A Forward Algorithm

---

**Algorithm A.1** Forward Algorithm.

---

**Input:**  $n \in \mathbb{N}$ . Markov source  $\mathcal{H}$  with state-space  $[\ell]$ , alphabet  $\Sigma$ , stationary distribution  $\pi \in \Delta([\ell])$ , transition matrix  $\Pi \in \mathbb{R}^{\ell \times \ell}$ , and output distributions  $\{\mathcal{S}_i \in \Delta(\Sigma)\}_{i \in [\ell]}$ . And  $y = (y_1, y_2, \dots, y_{n-1})$  for  $y_i \in \Sigma$ .

**Output:** Distribution  $Y_n \in \Delta(\Sigma)$

- 1: **procedure** FORWARDINFER( $\mathcal{H} = (\ell, \Sigma, \pi, \Pi, \{\mathcal{S}_i\}), n, y$ )
  - 2:    $s_0 \leftarrow \pi$ .
  - 3:   **for all**  $t = 1, 2, \dots, n - 1$  **do**
  - 4:     Define  $s_t \in \Delta([\ell])$  by  $s_t(i) \leftarrow \frac{(\Pi s_{t-1})_i \cdot \mathcal{S}_i(y_t)}{\sum_{j \in [\ell]} (\Pi s_{t-1})_j \cdot \mathcal{S}_j(y_t)}$  ▷ Treating  $s_{t-1}$  as a vector in the probability simplex embedded in  $\mathbb{R}^\ell$
  - 5:    $s_n \leftarrow \Pi s_{n-1}$ .
  - 6:   **Return** The distribution  $Y_n := \mathbb{E}_{i \sim s_n}[\mathcal{S}_i]$ .
- 

► **Claim 26.** For every Markov source  $\mathcal{H} = (\ell, \Sigma, \pi, \Pi, \{\mathcal{S}_i\})$ , let random variables  $Y_1, \dots, Y_n \sim \mathcal{H}_n$ . For every setting  $y = (y_1, y_2, \dots, y_{n-1})$  for  $y_i \in \Sigma$ , let  $\mathcal{D}_{Y_n | Y_{<n}=y}$  denote the distribution of  $Y_n$  conditioned on  $Y_{<n} = y$ . Then,

$$\text{FORWARDINFER}(\mathcal{H}, n, y) \equiv \mathcal{D}_{Y_n | Y_{<n}=y}$$

This follows inductively, from the fact that  $s_t$  as maintained by the algorithm is exactly the distribution of  $S_t | \{Y_{\leq t} = y_{\leq t}\}$ , where  $S_t$  is the hidden state of  $\mathcal{H}$  after  $t$  steps.

## B Connection to Learning Parity with Noise

The problem of learning parity with noise (LPN) is the following. Fix an (unknown) string  $a \in \mathbb{F}_2^\ell$  and  $\eta > 0$  and let  $D_{a,\eta}$  be the distribution on  $\mathbb{F}_2^{\ell+1}$  whose samples  $(x, y)$  are generated as follows: Draw  $x \in \mathbb{F}_2^\ell$  uniformly and let  $z \in \text{Bern}(\eta)$  be drawn independent of  $x$  and let  $y = \langle a, x \rangle + z$  where  $\langle a, x \rangle = \sum_{i=1}^\ell a_i x_i$ . Given samples  $(x_1, y_1), \dots, (x_m, y_m)$  drawn i.i.d. from such a distribution, the LPN problem is the task of “learning”  $a$ .



It is well known that  $a$  is uniquely determined by  $O(\ell)$  samples (i.e.,  $m = O(\ell)$ ) where the constant in the  $O(\cdot)$  depends on  $\eta < 1/2$ . However no polynomial time algorithms are known that work with  $m = \text{poly}(\ell)$  and determine  $a$  for any  $\eta > 0$  and indeed this is believed to be a hard task in learning. We refer to this hardness assumption as the LPN hypothesis.

The connection to learning Markovian sources comes from the fact that samples from the distribution  $D_{a,\eta}$  can be generated by an  $O(\ell)$ -state Markov chain. (Briefly the states are indexed  $(i, b, c)$  indicating  $\sum_{j=1}^{i-1} a_j x_j = b$  and  $x_i = c$ . For  $i < \ell$  the state  $(i, b, c)$  outputs  $c$  and transitions to  $(i+1, b+c, 0)$  w.p.  $1/2$  and to  $(i+1, b+c, 1)$  w.p.  $1/2$ . When  $i = \ell$ , the state  $(i, b, c)$  outputs  $(c, b+c)$  w.p.  $1-\eta$  and  $(c, b+c+1)$  w.p.  $\eta$  and transitions to  $(1, 0, 0)$  w.p.  $1/2$  and to  $(1, 0, 1)$  w.p.  $1/2$ .) The entropy of this source is  $(\ell + H(\eta))/(\ell + 1)$ . A compression with  $\varepsilon = (1 - H(\eta))/(2(\ell + 1))$  with  $\text{poly}(\ell/\varepsilon)$  samples from the source would distinguish this source from purely random strings which in turn enables recovery of  $a$ , contradicting the LPN hypothesis.

We thus conclude that compressing an *unknown* Markov source with number of samples that is a polynomial in the mixing time and the inverse of the gap to capacity contradicts the LPN hypothesis.



# On the Communication Complexity of Key-Agreement Protocols

**Iftach Haitner**<sup>1</sup>

The Blavatnik school of computer science, Tel Aviv University, Israel  
iftachh@cs.tau.ac.il

**Noam Mazor**<sup>2</sup>

The Blavatnik school of computer science, Tel Aviv University, Israel  
joanrpublic@dummyscollege.org

**Rotem Oshman**<sup>3</sup>

The Blavatnik school of computer science, Tel Aviv University, Israel  
rotem.oshman@gmail.com

**Omer Reingold**<sup>4</sup>

Computer Science Department, Stanford University, USA  
reingold@stanford.edu

**Amir Yehudayoff**<sup>5</sup>

Department of Mathematics, Technion-Israel Institute of Technology, Israel  
amir.yehudayoff@gmail.com

---

## Abstract

Key-agreement protocols whose security is proven in the *random oracle model* are an important alternative to protocols based on public-key cryptography. In the random oracle model, the parties and the eavesdropper have access to a shared random function (an “oracle”), but the parties are limited in the number of queries they can make to the oracle. The random oracle serves as an abstraction for black-box access to a symmetric cryptographic primitive, such as a collision resistant hash. Unfortunately, as shown by Impagliazzo and Rudich [STOC '89] and Barak and Mahmoody [Crypto '09], such protocols can only guarantee limited secrecy: the key of any  $\ell$ -query protocol can be revealed by an  $O(\ell^2)$ -query adversary. This quadratic gap between the query complexity of the honest parties and the eavesdropper matches the gap obtained by the *Merkle's Puzzles* protocol of Merkle [CACM '78].

In this work we tackle a new aspect of key-agreement protocols in the random oracle model: their *communication complexity*. In Merkle's Puzzles, to obtain secrecy against an eavesdropper that makes roughly  $\ell^2$  queries, the honest parties need to exchange  $\Omega(\ell)$  bits. We show that for protocols with certain natural properties, ones that Merkle's Puzzle has, such high communication is unavoidable. Specifically, this is the case if the honest parties' queries are uniformly random, or alternatively if the protocol uses non-adaptive queries and has only two rounds. Our proof for the first setting uses a novel reduction from the set-disjointness problem in two-party communication complexity. For the second setting we prove the lower bound directly, using information-theoretic arguments.

Understanding the communication complexity of protocols whose security is proven (in the random-oracle model) is an important question in the study of practical protocols. Our results and proof techniques are a first step in this direction.

---

<sup>1</sup> Supported by ERC starting grant 638121. Member of the Check Point Institute for Information Security.

<sup>2</sup> Supported by ERC starting grant 638121.

<sup>3</sup> Supported by the Israeli Centers of Research Excellence program 4/11 and BSF grant 2014256.

<sup>4</sup> Supported by NSF grant CCF-1749750.

<sup>5</sup> Supported by ISF grant 1162/15.



**2012 ACM Subject Classification** Theory of computation → Cryptographic protocols

**Keywords and phrases** key agreement, random oracle, communication complexity, Merkle’s puzzles

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2019.40

**Related Version** A full version of the paper is available at [8], <https://eccc.weizmann.ac.il/report/2018/031/>.

**Acknowledgements** We thank Yuval Ishai for challenging us with this intriguing question, and Omer Rotem for very useful discussions.

## 1 Introduction

In a key-agreement protocol [5], two parties communicating over an insecure channel want to securely agree on a shared secret key, such that an eavesdropper observing their communication cannot find the key. For example, given a hash function  $h : [n] \rightarrow [N]$  that is hard to invert, the players can execute the following protocol, called *Merkle’s puzzles* [13]: we fix an arbitrary parameter  $\ell \approx \sqrt{n}$ , and the parties select uniformly random subsets  $A = \{a_1, \dots, a_\ell\}$ ,  $B = \{b_1, \dots, b_\ell\} \subseteq [n]$  (respectively) of size  $\ell$ . We choose  $\ell, n$  such that with constant probability there is a unique intersection,  $|A \cap B| = 1$ . The first party evaluates  $h$  on every element  $a \in A$ , and sends  $h(a_1), \dots, h(a_\ell)$  to the second party, which then looks for a unique element  $b \in B$  such that  $h(b) = h(a_i)$  for some  $i \in [\ell]$ . If found, the second party sends the index  $i$  to the first party and outputs  $b$  as the secret key; the second party outputs  $a_i$  as the secret key. Because  $h$  is a “good” hash function and  $h(b) = h(a_i)$ , it is likely that  $b = a_i$ , so the players output the same key. Moreover, since  $h$  is hard to invert, an eavesdropper that tries to find the secret key after seeing  $h(a_1), \dots, h(a_\ell), i$  must essentially compute  $h$  on the entire universe in order to invert  $h$  and find  $a_i$ . Thus, we have a quadratic gap between the work performed by the eavesdropper, which must compute  $\Omega(\ell^2)$  hashes, and the work performed by the parties, which compute  $\ell$  hashes each.

Ideally we would strive for an *exponential* gap between the work required to break the security of the protocol and the work of the honest parties. There are numerous candidate constructions of such key-agreement schemes, e.g., [17, 14, 1, 12], based on assumptions implying that *public-key encryption schemes* exist. A fundamental open question is whether we can design key-agreement protocols based on the security of symmetric primitives (e.g., collision resistant hash); the security of such primitives is believed to be more robust than public-key encryption. A very important step in this direction was made by Barak and Mahmoody[2] (following Impagliazzo and Rudich[10]): they showed that as long as the symmetric primitive is used as a black box, the quadratic gap achieved by Merkle’s puzzles is the best possible.

The notion of “black box” is formalized by the *random oracle model*: instead of a concrete hash function  $h$ , we assume that the parties have access to a *random oracle*  $F : [n] \rightarrow [n]$ , a perfectly random function. The random oracle is “the best hash function possible” (w.h.p.), so lower bounds proven in the random oracle model hold for any instantiation where the oracle is replaced by a one-way function. Thus, the lower bound of Barak and Mahmoody rules out any black-box key-agreement scheme from one-way functions that achieves a better than quadratic gap between the eavesdropper’s work and the honest parties.

While a quadratic gap between the  $\ell$ -query honest parties and the  $\ell^2$ -query eavesdropper might not seem like much, and ideally we would wish for an exponential gap, on modern architecture it can yield a good enough advantage, assuming that security is preserved when the random oracle is replaced with a fixed hash function. For example, a consumer-level CPU (Intel Core i5-6600) can compute 5 million SHA-256 hashes per second, and specialized hardware for SHA-256 computation (for example, AntMiner S9) can compute  $14 \times 10^{12}$  hashes per second [3]. It follows that if the honest parties spend one second of computation on standard CPU, an attacker with specialized hardware can violate the security of Merkle's puzzles in less than a second. However, if the parties spend one second on specialized hardware, an attacker with specialized hardware has to spend more than 200,000 years to break the scheme.

So, are Merkle's puzzles a practical and realistic key-agreement scheme? The answer is probably not: even setting aside the question of replacing the random oracle by a concrete hash function, in Merkle's puzzles, the honest parties *send each other*  $\tilde{\Omega}(\ell)$  bits to obtain security against an eavesdropper that makes roughly  $\ell^2$  queries. In our example above, if we instantiate Merkle's puzzles using SHA-256 for one second on specialized hardware, the first party would need to send more than 100 terabytes to the second party. A fundamental question is whether this high communication burden is inherent to secure key-agreement, and more generally, what is the communication cost of cryptographic protocols in the random oracle model and other oracle models. In this paper we initiate the study of the communication complexity of cryptographic protocols in the random-oracle model.

## 1.1 Our Results

We show that for random-oracle protocols with certain natural properties, the high communication incurred by Merkle's puzzles is unavoidable: in order to achieve security against an adversary that can ask  $\Theta(\ell^2)$  queries, the two parties must exchange  $\Omega(\ell)$  bits of communication. Specifically, we show that the bound above holds for protocols where the parties' queries are a uniformly random set, and also for two-round protocols that make non-adaptive (but arbitrary) queries.<sup>6</sup> We stress that a general lower bound for non-adaptive key-agreement protocols would imply a lower bound on the communication complexity of the set-intersection problem. This fact suggests that it is unlikely to find a simple proof for the general case.

To simplify the statements of our results, we focus here on key-agreement protocols whose *agreement* parameter, the probability that the players output the same key, is larger by some constant than their *secrecy* parameter, the probability that an eavesdropper can find the key.

### Uniform-query protocols.

We say that a random-oracle protocol makes *uniform queries* if each party's oracle queries are a uniformly random set. We give the following lower bound on the communication complexity of such protocols.

► **Theorem 1** (lower bound on uniform-queries protocols, informal). *Any  $\ell$ -uniform-query key-agreement protocol achieving non-trivial secrecy against  $o(\ell^2)$ -query adversaries has communication complexity  $\Omega(\ell)$ .*

This theorem is proved by a reduction from *set-disjointness*, a problem in communication complexity that is known to require high communication.

<sup>6</sup> These are both properties of Merkle's puzzles.

**Two-round non-adaptive protocols.**

An oracle protocol is said to make *non-adaptive* queries if the distribution of queries made by the players is fixed in advance, i.e., it is determined before the parties communicate with each other and does not depend on the oracle’s answers. We give the following lower bound on the communication complexity of such protocols.

► **Theorem 2** (lower bound on two-message non-adaptive protocols, informal). *Any two-message  $\ell$ -query non-adaptive key-agreement protocol of non-trivial secrecy against  $q$ -query adversaries has communication complexity  $\Omega(q/\ell)$ .*

Once again this lower bound is nearly-tight with Merkle’s puzzles, where  $q = \Theta(\ell^2)$ , and the communication cost is  $\tilde{\Theta}(\ell)$ .<sup>7</sup>

Following Barak and Mahmoody [2] and Impagliazzo and Rudich [10]), we prove this lower bound by presenting an eavesdropper that makes  $q$  queries and prevents the parties from exploiting the advantage they gain by their joint random oracle calls.

In [2], the communication cost of the protocol is not taken into account: their eavesdropper makes  $O(\ell^2)$  queries and has high probability of finding *all* intersection queries (i.e., all queries that were asked by both players). In our case, if the protocol has communication cost  $C$ , then to prove Theorem 2, our eavesdropper must make only  $O(C \cdot \ell)$  queries (to show the trade-off that  $C = \Omega(q/\ell)$ ). If  $C \ll \ell$ , our eavesdropper makes much fewer queries than the eavesdropper in [2, 10], and in particular it cannot discover all the intersection queries. Instead, our eavesdropper asks only queries that the players *were able to learn* are in their intersection. If a query is in the intersection, but the players have not communicated this fact to each other, then the eavesdropper will not necessarily ask this query (unlike [2, 10]). Finding the correct definition for what it means to “learn” that a given query is in the intersection, and constructing an eavesdropper that makes only  $O(C \cdot \ell)$  queries, are the main difficulty in our proof.

**1.2 Related Work**

Impagliazzo and Rudich [10] showed that the key of any  $\ell$ -query key-agreement protocol in the random-oracle model can be revealed by an  $\tilde{O}(\ell^6)$  query eavesdropper. Barak and Mahmoody [2] improve this bound and present an  $O(\ell^2)$  query eavesdropper for this task, which shows that Merkle puzzles is optimal in this respect. Haitner, Omri, and Zarusim [9] used the machinery of [2] to relate the security of protocols that do not use a random oracle and solve tasks with no input, to the security of no-input protocols in the random-oracle model against an  $O(\ell^2)$ -query adversary. Finding limitations on the usefulness of random oracles for protocols that do take input seems to be a more difficult question. Chor and Kushilevitz [4] and Mahmoody et al. [11] made some progress in this direction. Finally, Haitner, Hoch, Reingold, and Segev [7] gave lower bounds on the communication complexity of statistically hiding commitments and single-server private information retrieval in a weaker oracle model that captures the hardness of one-way functions/permutation more closely than the random-oracle model.

<sup>7</sup> This theorem is also nearly-tight for any  $q$ , with a version of Merkle’s puzzle, in which Alice is sending  $\Theta(q/l)$  answers, from a universe of size  $\Theta(q)$ .

## 1.3 Organization

We begin by giving the formal definitions and notation used throughout the paper in Section 2. High-level overview of our proof techniques is given in Sections 3 and 4. For full proofs, see the full version of this paper in [8].

## 2 Preliminaries

### 2.1 Notations

We use calligraphic letters to denote sets, uppercase for random variables and lowercase for values. For  $m \in \mathbb{N}$ , let  $[m] = \{1, \dots, m\}$ . For a random variable  $X$ , let  $x \stackrel{R}{\leftarrow} X$  to denote that  $x$  is chosen according to  $X$ . Similarly, for a set  $S$  let  $s \stackrel{R}{\leftarrow} S$  to denote that  $s$  is chosen according to the uniform distribution over  $S$ . The support of the distribution  $D$ , denoted  $\text{Supp}(D)$ , is defined as  $\{u \in \mathcal{U} : \Pr_D[u] > 0\}$ . The statistical distance between two distributions  $P$  and  $Q$  over a finite set  $\mathcal{U}$ , denoted  $\text{SD}(P, Q)$ , is defined as  $\frac{1}{2} \sum_{u \in \mathcal{U}} |\Pr_P[u] - \Pr_Q[u]|$ , which is equal to  $\max_{S \subseteq \mathcal{U}} (\Pr_P[S] - \Pr_Q[S])$ .

For a vector  $\vec{X} = X_1, \dots, X_n$  and an index  $i \in [n]$ , let  $X_{<i}$  denote the vector  $X_1, \dots, X_{i-1}$  and  $X_{\leq i}$  denote the vector  $X_1, \dots, X_i$ . For a set of indexes  $T = \{i_1, \dots, i_k\} \subseteq [n]$  such that  $i_1 < i_2 < \dots < i_k$ , let  $X_T$  denote the vector  $X_{i_1}, \dots, X_{i_k}$ . Similarly,  $X_{T, <i}$  denotes the vector  $X_{T \cap \{1, \dots, i-1\}}$ . For a function  $f$ , let  $f(\vec{X}) = (f(X_1), \dots, f(X_n))$ .

For random variables  $A$  and  $B$  we use  $A|_{B=b}$  to denote the distribution of  $A$  condition on the event  $B = b$ , and  $A \times B$  to denote the product between the marginal distributions of  $A$  and  $B$ . When  $A$  is independent from  $B$  we write  $A \perp B$  to emphasize that this is the case.

### 2.2 Interactive Protocols

A two-party protocol  $\Pi = (A, B)$  is a pair of probabilistic interactive Turing machines. The communication between the Turing machines  $A$  and  $B$  is carried out in rounds, where in each round one of the parties is active and the other party is idle. In the  $j$ -th round of the protocol, the currently active party  $P$  acts according to its partial view, writing some value on its output tape, and then sending a message to the other party (i.e., writing the message on the common tape). The communication transcript (henceforth, the transcript) of a given execution of the protocol  $\Pi = (A, B)$ , is the list of messages  $m$  exchanged between the parties in an execution of the protocol, where  $m_{1, \dots, j}$  denotes the first  $j$  messages in  $m$ . A view of a party contains its input, its random tape and the messages exchanged by the parties during the execution. Specifically,  $A$ 's view is a tuple  $v_A = (i_A, r_A, m)$ , where  $i_A$  is  $A$ 's input,  $r_A$  are  $A$ 's random coins, and  $m$  is the transcript of the execution. Let  $\text{out}^A$  denote the output of  $A$  in the end of the protocol, and  $\text{out}^B$   $B$ 's output. Notice that given a protocol, the transcript and the outputs are deterministic function of the joint view  $(i_A, r_A, i_B, r_B)$ . For a joint view  $v$ , let  $\text{trans}(v)$ ,  $\text{out}^A(v)$  and  $\text{out}^B(v)$  be the transcript of the protocol and the parties' outputs determined by  $v$ . For a distribution  $D$  we denote the distribution over the parties' joint view in a random execution of  $\Pi$ , with inputs drawn from  $D$  by  $\Pi(D)$ .

A protocol  $\Pi$  has  $r$  rounds, if for every possible random tapes for the parties, the number of rounds is exactly  $r$ . The Communication Complexity of a protocol  $\Pi$ , denoted as  $\text{CC}(\Pi)$  is the length of the transcript of the protocol in the worst case.



### 2.3 Oracle-Aided Protocols

An oracle-aided two-party protocol  $\Pi = (A, B)$  is a pair of interactive Turing machines, where each party has an additional tape called the oracle tape; the Turing machine can make a query to the oracle by writing a string  $q$  on its tape. It then receives a string *ans* (denoting the answer for this query) on the oracle tape. An oracle-aided protocol is  $\ell$ -queries protocol if each party makes at most  $\ell$  queries during each run of the protocol. In a *non-adaptive* oracle-aided protocol, the parties choose their queries before the protocol starts and before querying the oracle. A *uniform query* oracle-aided protocol, is a non-adaptive protocol in which the parties queries are chosen uniformly from a predetermined set.

### 2.4 Key-Agreement Protocols

Since we are giving lower bounds, we focus on single bit protocols.

► **Definition 3** (key-agreement protocol). Let  $0 \leq \gamma, \alpha \leq 1$  and  $q \in N$ . A two-party boolean output protocol  $\Pi = (A, B)$  is a  $(q, \alpha, \gamma)$ -key-agreement relative to a function family  $\mathcal{F}$ , if the following hold:

**Accuracy:**  $\Pi$  has  $(1 - \alpha)$ -accuracy. For every  $f \in \mathcal{F}$ :

$$\Pr_{v \stackrel{R}{\leftarrow} \Pi^f} [\text{out}^A(v) = \text{out}^B(v)] \geq 1 - \alpha.$$

**Secrecy:**  $\Pi$  has  $(q, \gamma)$ -secrecy. For every  $q$ -query oracle-aided algorithm  $E$ :

$$\Pr_{f \stackrel{R}{\leftarrow} \mathcal{F}, v \stackrel{R}{\leftarrow} \Pi^f} [E^f(\text{trans}(v)) = \text{out}^A(v)] \leq \gamma.$$

If  $\mathcal{F}$  is a trivial function family (e.g.,  $\mathcal{F}$  contains only the identity function), then all correlation between the parties' view is implied by the transcript. Hence, an adversary that on a given transcript  $\tau$  samples a random view for  $A$  that is consistent with  $\tau$ , and outputs whatever  $A$  would upon this view, agrees with  $B$  with the same probability as does  $A$ . This simple argument yields the following fact:

For every  $0 \leq \alpha \leq 1$  and  $0 \leq \gamma < 1 - \alpha$ , there exists no  $(q, \alpha, \gamma)$ -key-agreement protocol relative to the trivial family.

### 2.5 Entropy and Information

In the proof we often need to measure differences between various distributions. For this purpose we use *f-divergences*: given a convex function  $f : \mathbb{R} \rightarrow \mathbb{R}$  with  $f(1) = 0$ , and distributions  $P, Q$ , the *f-divergence of P from Q* is defined as

$$D_f(P \parallel Q) = \sum_{q \in \mathcal{Q}} \Pr[Q = q] f\left(\frac{\Pr[P = q]}{\Pr[Q = q]}\right).$$

Specifically, the two *f-divergences* we use in this paper are the *statistical distance*, obtained by taking  $f(x) = |x - 1|/2$ , and the *KL divergence*, obtained by taking  $f(x) = x \log x$ . Each has its own nice properties and disadvantages: statistical distance is bounded in  $[0, 1]$  but it is not additive, while KL divergence is additive but unbounded.

We frequently need to measure the “amount of dependence” between two random variables. Let  $(X, Y) \sim P_{X,Y}$  be random variables jointly distributed according to  $P_{X,Y}$ , and let  $P_X, P_Y$  be the marginal distribution of  $X$  and  $Y$ , respectively. Also, let  $P_X \times P_Y$  be the product distribution where  $X$  and  $Y$  are sampled independently of each other, each from its marginal

distribution  $P_X, P_Y$  (respectively). To quantify the dependence between  $X$  and  $Y$ , we measure the difference between their joint distribution and the product of the marginals: formally, we define

$$I_f(X; Y) = D_f(P_{X,Y} \parallel P_X \times P_Y).$$

This generalizes the usual notion of mutual information, which is the special case of  $I_f$  where we use KL divergence (i.e., when  $f = x \log x$ ). For clarity, when we use KL divergence we omit the subscript  $f$ , and when using statistical distance, we use the notation  $I_{SD}$  (instead of  $I_{f(x)=|x-1|/2}$ ).

Finally, we also need the notion of *conditional mutual information*, which is simply the average mutual information between two variables  $X, Y$ , where the average is taken over a third random variable  $Z$ . Formally, let  $(X, Y, Z) \sim P_{X,Y,Z}$ . For any value  $z$ , let  $P_{X,Y|Z=z}, P_{X|Z=z}, P_{Y|Z=z}$  be the joint distribution of  $X, Y$  and the marginals of  $X$  and  $Y$ , respectively, all conditioned on the event  $Z = z$ .

Then we define  $I_f(X; Y|Z) = \mathbb{E}_{z \sim P_Z} [D_f(P_{X,Y|Z=z} \parallel P_{X|Z=z} \times P_{Y|Z=z})]$ .

In the next sections we outline the strategy of the proofs.

### 3 Uniform-Query Protocols: Proof Outline

Our lower bound for uniform-query key-agreement protocols is proved via a reduction to *set disjointness*, a classical problem in two-party communication complexity.

In the set disjointness problem, we have two players, Alice and Bob. The players receive inputs  $X, Y \subseteq [n]$ , respectively, of size  $|X| = |Y| = \ell$ , and the players must determine whether  $X \cap Y = \emptyset$ . To do this, the players communicate with each other, and the question is how many bits they must exchange. It is known [16] that for any sufficiently large  $n \in \mathbb{N}$ , if the size of the sets is  $\ell = n/4$ , then the players must exchange  $\Omega(n)$  bits to solve set disjointness, and this holds even for randomized protocols where the players have access to shared randomness and only need to succeed with probability  $2/3$ . Here, we require high success probability *on any input*, not over some specific input distribution. We note that in the 2-party communication complexity model there is no random oracle.

The connection between set disjointness and key agreement comes from the fact that the only *correlation* between the parties' views in a key agreement protocol comes from the *intersection queries*, the queries that both players ask and Eve does not know. Indeed, if Alice asks  $A \subseteq [n]$  and Bob asks  $B \subseteq [n]$ , and the random oracle is  $F : [n] \rightarrow [n]$ , then  $F(A \setminus (A \cap B))$  and  $F(B \setminus (A \cap B))$  are *independent of each other*. In particular, if  $A \cap B = \emptyset$ , then  $F(A)$  and  $F(B)$  are independent, and intuitively, in this case the players cannot securely agree on a secret key, because they have no advantage over the eavesdropper. On the other hand, if  $A \cap B \neq \emptyset$ , then the players can exploit the correlation induced by  $F(A \cap B)$  to securely agree on a secret key. Thus, any secure key agreement protocol “behaves differently” depending on whether  $A \cap B = \emptyset$  or not, and we can use this to solve the set disjointness problem.

Suppose that we are given a secure key-agreement protocol  $\Pi$ , where the players make  $\ell$  uniformly-random queries to an oracle  $F : [n] \rightarrow [n]$ . For simplicity we assume that the protocol has *perfect agreement*, that is, the players always output the same key, and that the security parameter is  $3/4$ , that is, an eavesdropper has probability at most  $3/4$  of outputting the same key as the players. Our full proof does not make these assumptions.

Now, we want to construct from the key-agreement protocol  $\Pi$ , which uses a random oracle, a protocol  $\Pi'$  for set disjointness, *without* a random oracle (as usual in communication complexity). To this end, we consider two possible ways of simulating  $\Pi$  without an oracle:

## 40:8 The Communication Complexity of Key-Agreement

- $\Lambda_{\text{Com}}$ : the players use their shared randomness to simulate the oracle. They interpret the shared randomness as a random function  $F : [n] \rightarrow [n]$ , and whenever  $\Pi$  wants to query some element  $q \in [n]$ , the players use  $F(q)$  as the oracle's answer.
- $\Lambda_{\text{Dist}}$ : the players use their *private* randomness to simulate the oracle. Alice and Bob interpret their private randomness as random functions  $F_A, F_B : [n] \rightarrow [n]$ , respectively. Whenever  $\Pi$  indicates that Alice should query an element  $q \in [n]$ , she uses  $F_A(q)$  as the answer, while Bob uses  $F_B(q)$ .

The first simulation,  $\Lambda_{\text{Com}}$ , is “perfect”: it produces exactly the correct distribution of transcripts and outputs under our key-agreement protocol  $\Pi$ . In particular, the keys produced by the players in  $\Lambda_{\text{Com}}$  always agree, and an eavesdropper that sees the transcript of  $\Lambda_{\text{Com}}$  (but not the shared randomness) can find the key with probability at most  $3/4$ .

On the other hand, the second simulation  $\Lambda_{\text{Dist}}$  is “wrong”, because the players do not use the same random function to simulate the random oracle. In fact, it is known that without shared randomness, secure key agreement is *impossible*, as an eavesdropper that sees the transcript can find the key with the same probability that the players have of agreeing with each other. Therefore there are two possible cases:

**Agreement gap:** The probability that the players agree on the key in  $\Lambda_{\text{Dist}}$  is at most  $7/8$  (compared to one in  $\Lambda_{\text{Com}}$ ), or

**Secrecy gap:** There is an eavesdropper  $E$  that guesses Alice's key in  $\Lambda_{\text{Dist}}$  with probability at least  $7/8$  (compared to  $3/4$  in  $\Lambda_{\text{Com}}$ ).

(Instead of  $7/8$  we could have used here any constant probability in  $(3/4, 1)$ , but in the full proof this choice depends on the agreement and security parameters of  $\Pi$ .)

We divide into cases, depending on which of the two gaps we have.

### Agreement gap

Assume that the players agree with probability at most  $7/8$  in  $\Lambda_{\text{Dist}}$ . For simplicity, let us make the stronger assumption that for any intersection size  $c > 0$ , the probability of agreement between the players is at most  $7/8$ , even *conditioned* on the event that  $|A \cap B| = c$ . A general key-agreement protocol might not satisfy this assumption, which complicates the full proof significantly; see the full version of this paper for the details.

So, we assumed that whenever the intersection is non-empty, the players agree with probability at most  $7/8$ . Observe, however, that when the intersection *is* empty ( $A \cap B = \emptyset$ ), the distribution of transcript and outputs in  $\Lambda_{\text{Dist}}$  is the same as in  $\Pi$ : although each player uses a different random function, they never ask the same query, so there is no inconsistency. Therefore, conditioned on  $A \cap B = \emptyset$ , in  $\Lambda_{\text{Dist}}$  the players have perfect agreement (as in  $\Pi$ ). In other words,  $\Lambda_{\text{Dist}}$  behaves very differently when  $A \cap B = \emptyset$ , in which case the players always agree on the key, compared to the general case, where the players agree with probability at most  $7/8$ . We use this fact to *check* whether  $A \cap B = \emptyset$ . Thus, by checking whether or not they got the same key in  $\Lambda_{\text{Dist}}$ , the players get an indication for whether or not  $A \cap B = \emptyset$ .

Our set disjointness protocol  $\Pi'$  is defined as follows. Given inputs  $X, Y \subseteq [n]$ , respectively, the players simulate  $\Lambda_{\text{Dist}}$  several times. In each simulation, the players agree on a random permutation  $\sigma : [n] \rightarrow [n]$  using their shared randomness, and then the players simulate  $\Lambda_{\text{Dist}}$  using their permuted inputs as the query set; that is, Alice feeds  $A = \sigma(X)$  to  $\Lambda_{\text{Dist}}$  as her query set, and Bob feeds  $B = \sigma(Y)$  to  $\Lambda_{\text{Dist}}$  as his query set. Note that  $A, B$  are uniformly random, *subject to* having an intersection of size  $|X \cap Y|$ .

After each simulation of  $\Lambda_{\text{Dist}}$ , the players send each other the keys output under  $\Lambda_{\text{Dist}}$ , and check if they got the same key. Finally, they output “ $X \cap Y = \emptyset$ ” iff they got the same key in all the simulations of  $\Lambda_{\text{Dist}}$ .

Since  $\Lambda_{\text{Dist}}$  has perfect agreement when there is no intersection, the players always succeed when  $X \cap Y = \emptyset$ . However, by assumption, whenever  $X \cap Y \neq \emptyset$ , the probability of agreement in  $\Lambda_{\text{Dist}}$  is at most  $7/8$ , so if we repeat  $\Lambda_{\text{Dist}}$  sufficiently many times, the probability that all instances output the same key will be at most  $1/3$ .

### Secrecy gap

In this case we convert  $\Lambda_{\text{Com}}$  and  $\Lambda_{\text{Dist}}$  into a pair of protocols with an agreement gap, and then proceed as above.

Consider the protocol  $\Lambda'_{\text{Dist}}$  where the parties acts as in  $\Lambda_{\text{Dist}}$ , but at the end, Bob executes the eavesdropper E on the transcript, and outputs the key that E outputs. Define  $\Lambda'_{\text{Com}}$  analogously.

By assumption, E guesses Alice's output in  $\Lambda_{\text{Dist}}$  with probability at least  $7/8$ , but in  $\Lambda_{\text{Com}}$  it succeeds with probability at most  $3/4$ . Thus, in  $\Lambda'_{\text{Dist}}$  the players agree with probability at least  $7/8$ , but in  $\Lambda'_{\text{Com}}$  they agree with probability at most  $3/4$ ; there is a gap of at least  $1/8$  between the probability of agreement in the two protocols (although they have switched roles and now  $\Lambda'_{\text{Dist}}$  has the higher agreement probability). Note also that  $\Lambda'_{\text{Dist}}$  does not have agreement probability 1, as we assumed for simplicity above, but our full proof can handle this case.

### Formal statement

Here we give the formal statements that we prove for uniform-query protocols. Formally, our reduction is to set-disjointness over the distribution below, which known to be hard for low complexity protocols.

► **Definition 4** (hard distribution for set-disjointness). For  $\ell \in \mathbb{N}$ , let  $\mathcal{Q}_\ell^0 = \{\mathcal{X}, \mathcal{Y} \subset [\ell]: |\mathcal{X}| = |\mathcal{Y}| = \lfloor \ell/4 \rfloor, \mathcal{X} \cap \mathcal{Y} = \emptyset\}$  and let  $\mathcal{Q}_\ell^1 = \{\mathcal{X}, \mathcal{Y} \subset [\ell]: |\mathcal{X}| = |\mathcal{Y}| = \lfloor \ell/4 \rfloor, |\mathcal{X} \cap \mathcal{Y}| = 1\}$ . Let  $D_\ell^0$  and  $D_\ell^1$  be the uniform distribution over  $\mathcal{Q}_\ell^0$  and  $\mathcal{Q}_\ell^1$  respectively, and let  $D_\ell = \frac{3}{4} \cdot D_\ell^0 + \frac{1}{4} \cdot D_\ell^1$ .

Razborov [16] has shown that solving set-disjointness  $D_\ell$  with small error require high communication complexity.

► **Theorem 5** (hardness of  $D_\ell$ , [16]). *Exists  $\epsilon > 0$  such that for every  $\ell \in \mathbb{N}$  and a protocol  $\Pi$  that solves set-disjointness over  $D_\ell$  with error  $\epsilon$ , it holds that  $\text{CC}(\Pi) \geq \Omega(\ell)$ .*

For a finite set  $\mathcal{S}$ , let  $\mathcal{F}_\mathcal{S} = \{f : \mathcal{S} \mapsto \{0, 1\}^*\}$  be the family of all functions from  $\mathcal{S}$  to binary strings. Our reduction is stated in the following theorem.

► **Theorem 6** (from uniform-query key-agreement protocols to set-disjointness). *Assume exists an  $\ell$ -uniform-query  $(0, \alpha, \gamma)$ -key agreement protocol relative to  $\mathcal{F}_\mathcal{S}$ , for some set  $\mathcal{S}$ , of communication complexity  $c$ . Then there exists a protocol for solving set-disjointness over  $D_\ell$  with  $\epsilon$  error and communication complexity  $\frac{2^{15} \cdot \ell^4 \cdot \log 1/\epsilon}{|\mathcal{S}|^2 (1 - \alpha - \gamma)^4} \cdot c$ .*

Note that the above theorem holds also for protocols that are only secure against eavesdropper without access to the oracle. Combining theorems 5 and 6 yields the following bound on the communication complexity of uniform-query key-agreement protocols.

► **Theorem 7** (Main result for uniform-inputs protocols). *For any  $\ell$ -uniform-query  $(q, \alpha, \gamma)$ -key agreement protocol  $\Pi$  relative to  $\mathcal{F}_\mathcal{S}$ , it holds that  $\text{CC}(\Pi) \in \Omega((1 - \alpha - \gamma)^4 q^2 / \ell^3)$ .*

### What about general protocols?

It was important for our reduction to assume that the key-agreement protocol makes uniformly-random queries. Indeed, this reduction fails in the general case: consider the protocol where Alice and Bob always query 1, and output  $F(1)$  as their secret key. This protocol is completely insecure, since the eavesdropper can also query 1 and output  $F(1)$ . But our reduction would not work for it, because the input distribution where both players get the set  $\{1\}$  is not hard for set disjointness (indeed it is trivial). We see that the “hardness” of secure key-agreement is not necessarily that it is hard for the players to find their intersection queries, but that the eavesdropper should not be able to *predict* the intersection queries that the players use. Our second lower bound makes this intuition explicit and uses it to get a lower bound on two-round protocols with arbitrary (but non-adaptive) query distributions.

## 4 Two-Message Non-Adaptive Protocols: Proof Outline

In this section we describe a lower bound on the communication cost of any key-agreement protocol that makes non-adaptive queries and uses two rounds of communication: we show that any such protocol that makes  $\ell$  queries and is secure against an adversary that makes  $q$  queries must send a total of  $\Omega(q/\ell)$  bits. In particular, taking  $q = \Theta(\ell^2)$ , this shows that Merkle’s puzzles is optimal in its communication cost. Formally, we show the following bound, where  $\mathcal{F}_n$  is the family of all functions from  $\{0, 1\}^n$  to  $\{0, 1\}^n$ :

► **Theorem 8** (Main theorem for two-message, non-adaptive protocols). *For any  $n \in \mathbb{N}$ , the communication complexity of a two-message, non-adaptive,  $\ell$ -query  $(q, \alpha, \gamma)$ -key-agreement protocol relative to  $\mathcal{F}_n$  is at least*

$$\frac{(1 - \alpha - \gamma)^2 q}{50^2 \ell} - 6.$$

In this proof, we once again relate the parties’ advantage over the eavesdropper to the information they gained about the intersection of their query sets. We show that to produce a shared key, the parties need to learn a lot of information about this intersection. Moreover, the query sets and their intersection need to be “unpredictable” (have high min-entropy) given the transcript, otherwise an eavesdropper could make the same queries and output the same key.

### 4.1 Some examples

Let us illustrate the ideas behind the lower bound by way of some examples.

#### Example 1

We already discussed the naive example where both players query 1 and output  $F(1)$ , and said that it is insecure because the eavesdropper can *predict* the intersection query. Here is another instantiation of this idea: Alice and Bob view the domain  $\ell^2$  as an  $\ell \times \ell$  matrix, so that the oracle queries are represented by pairs  $(i, j) \in [\ell]^2$ . Alice chooses a row  $a \in [\ell]$ , and queries all the elements of the row (that is, all pairs  $(a, j)$  where  $j \in [\ell]$ ); Bob chooses a column  $b \in [\ell]$  and queries all the elements of the column (all pairs  $(i, b)$  where  $i \in [\ell]$ ). Then, Alice sends  $a$  to Bob, who responds with  $F(a, b)$ . From  $F(a, b)$ , Alice can compute  $b$ , by finding the (w.h.p. unique) index  $j$  such that  $F(a, b) = F(a, j)$ . Both players output the first bit of  $b$  as the key.

This protocol is slightly less naïve than the previous one: now there are no queries that have high prior probability of being asked, and the index  $b$  of the query that determines the key is uniformly random a-priori. However, once Alice sends  $a$  to Bob, the game is up: Eve can also query row  $a$  and find  $b$  the same way Alice does.

We see that in addition to queries that have a high prior probability of being asked, Eve also needs to ask queries that have a high *posterior* probability of being asked, after she sees  $M_1$ . It turns out that this is enough: if we were to continue for more than 2 rounds, then Eve would also need to ask queries that become likely after seeing  $M_2$ , and so on, but to prove a 2-round lower bound, Eve does not need to ask these queries. Intuitively, if a query only becomes likely after  $M_2$  is sent, then this is “too late” for it to be useful to the players, and Eve can ignore it.

## Example 2

First, both players query 1. Then they carry out the protocol from Example 1, but all messages are “encrypted” by XOR-ing them with  $F(1)$ .

From this example we see that Eve needs to be somewhat adaptive: when she decides what queries to ask after seeing  $M_1$ , she must incorporate the queries she asked before the first round (in this case, she would query 1). Essentially, when Eve tries to understand what the players have done in round  $i$ , she should take into account all the queries she made up to round  $i$ .

Should Eve be adaptive *inside* each round? In other words, after seeing  $M_1$ , should she ask all queries  $\mathcal{E}_1$  that became likely, then compute which new queries are now likely given  $M_1, \mathcal{E}_1$ , and so on, until she reaches a fixpoint?

It turns out that for our purposes here, because we consider non-adaptive protocols, Eve does not need to do this.

## Heavy queries

Our attacker Eve tries to break the security of the protocol by asking all queries that are “somewhat likely” to be asked by the players; these queries are called *heavy queries*. Informally, a query  $q \in \{0, 1\}^n$  is *heavy* after round  $i$  if given the transcript up to round  $i$  (inclusive), and given Eve’s queries up to round  $i$ , the probability that  $q$  is asked by one (or both) of the players exceeds some threshold  $\delta$  which is fixed in advance.

More formally, the set  $\mathcal{E}_i$  of heavy queries after round  $i$  is defined by induction on rounds, as follows: the a-priori heavy queries,  $\mathcal{E}_0$ , are given by

$$\mathcal{E}_0 = \{q \in \{0, 1\}^n : \Pr[q \in X \cup Y] \geq \delta\}.$$

These are queries that are “somewhat likely” to be asked before the protocol begins. For  $i > 0$ , we define

$$\mathcal{E}_i = \mathcal{E}_{i-1} \cup \{q \in \{0, 1\}^n : \Pr[q \in X \cup Y \mid M_{\leq i}, F(\mathcal{E}_{i-1})] \geq \delta\}.$$

In other words, after round  $i$ , Eve asks all queries  $q \in \{0, 1\}^n$  that have probability at least  $\delta$  of being queried by the players, given the messages  $M_{\leq i}$  that Eve observed up to round  $i$  and the heavy queries she asked before,  $\mathcal{E}_{i-1}$ .

### A simplified normal form for protocols

To simplify the proof of the lower bound, we first apply an easy transformation to the protocol: given a key agreement protocol  $\Pi$ , we construct a protocol  $\Pi'$ , which has nearly the same communication and query complexity as  $\Pi$ , the same number of rounds, and the same agreement and nearly the same security parameters. But  $\Pi'$  also has the following properties: first,  $\Pi'$  has no a-priori heavy queries, that is,  $\mathcal{E}_0 = \emptyset$ ; and second, the secret key output by Bob in  $\Pi'$  is the first bit of Bob's last query. This easy transformation is omitted in this overview.

### Measuring the Players' Advantage Over Eve

As we saw in the examples above, the players' ability to produce a shared secret key is closely tied to how much information *the players* have that *Eve does not have* about the intersection of the query sets,  $X \cap Y$ .

To quantify this advantage, define the following random variables:

- $S_i = (X \cap Y) \setminus \mathcal{E}_{i-1}$ , the intersection queries that have not been asked by Eve.
- $F(S_i)$ : the answers to the queries in  $S_i$ .
- $V_E^i = (\mathcal{E}'_i, F(\mathcal{E}'_i))$ : a subset of the heavy queries for the previous round, and the answers to them. Here,  $\mathcal{E}'_i \subseteq \mathcal{E}_i$  is a subset that will be defined later (and depends on the round number  $i$ ). For technical reasons, it is convenient to use only some of the heavy queries in some contexts; essentially, in some places in our proof, Eve uses only some of her power. This helps us avoid some unnecessary dependencies.

We measure the advantage gained by the players in round  $i$  by an expression of the form:

$$I_f(S_i, F(S_i); M_i | Z, V_E^{i-1}, M_{<i}), \quad (1)$$

where  $I_f$  is the information with respect to the  $f$ -divergence,  $M_i, M_{<i}$  are the  $i$ -th message and the messages of rounds  $1, \dots, i-1$ ,  $Z$  is the query set of the player that sent  $M_i$  (either  $X$  or  $Y$ , depending on the round number  $i$ ). Note that Eve uses her heavy queries from the previous rounds,  $V_E^{i-1}$ , to “try to understand” what is going on in the current round.

Intuitively, this expression measures how much information the  $i$ -th message conveys about the intersection queries and their answers, which Eve *cannot guess*. For this reason, the random variable  $S_i$  excludes intersection queries that were asked by Eve. Notice that on the right-hand side we condition on Eve's view (or on things Eve can sample): Eve has already seen the messages  $M_{<i}$  and asked the heavy queries  $V_E^{i-1} = (\mathcal{E}'_{i-1}, F(\mathcal{E}'_{i-1}))$ , and she can sample the queries  $Z$ , either  $X$  or  $Y$ , from the correct distribution given the transcript and her queries. Crucially, this does not require her to make any oracle queries: we do not require her to sample the answers  $F(Z)$ , only the queries  $Z$ . In other words, Eve can *pretend* to be whichever player the query set  $Z$  belongs to, and by conditioning on her view, we essentially neutralize all the information that Eve can extract about the intersection. Thus, the expression in (1) measures the information the players gain about the intersection but that is hidden from Eve.<sup>8</sup>

Our proof consists of showing:

<sup>8</sup> As we said above, we use only some of Eve's heavy queries,  $\mathcal{E}'_{i-1} \subseteq \mathcal{E}_i$ , so this intuition is not completely accurate; specifically, when (1) is *large*, it does not mean that Eve cannot guess a lot about the intersection, because she could use the full set  $\mathcal{E}_i$ . However, when (1) is *small*, then indeed Eve knows almost as much about the intersection as the players do, because her view *includes*  $V_E^{i-1}$  (and possibly more).



- Step I:** After the first message  $M_1$  is sent, the advantage gained is small, only  $O(\delta|M_1|)$ . For this part of the proof we use KL-divergence to measure the advantage.
- Step II:** After the second message  $M_2$  is sent, the advantage is still small, only  $O(\sqrt{\delta(|M_1| + |M_2|)})$ . Here we use statistical distance to measure the advantage, for reasons we will explain below.
- Step III:** When the expression in (1) is small (i.e., the players only have a small “advantage”), then indeed, Eve can break the security of the protocol, by pretending to be one of the players and sampling the secret key that this player would output.

Next we explain in more detail how each step is carried out.

## 4.2 Outline of the Proof

### Step III: How Eve breaks security

Let us start from the end: suppose that after the second round, the “advantage” is small:

$$I_f(S_2, F(S_2); M_2 | Y, M_1, V_E^1) \leq \beta,$$

where  $\beta = O(\sqrt{\delta(|M_1| + |M_2|)}) \ll 1$ . Here, the advantage is measured in statistical distance (that is, we take  $f(t) = |t - 1|/2$ ). We want to show that Eve can break the security of the protocol, by guessing the secret key.

As we said, Eve’s strategy is to “pretend” that she is Bob, and sample Bob’s output,  $\text{out}^B$ .

In a general protocol, to do this, Eve needs to sample Bob’s queries  $Y$  and the answers  $F(Y)$ , and then she can compute  $\text{out}^B = \text{out}^B(Y, F(Y), M_1, M_2)$ . However, recall that we transformed the protocol so that  $\text{out}^B$  is a fixed function of  $Y$ ; therefore, Eve in fact needs to do nothing clever, only sample  $Y$  given her view  $M_1, M_2, \mathcal{E}_1, F(\mathcal{E}_1)$  and compute  $\text{out}^B$  from  $Y$ .

We need to show that Eve’s key is close to the correct distribution, the one used by the players. In general, if too much communication is allowed, this is not true, as shown by the following example.

► **Example 9.** In Merkle’s puzzles, Alice’s message is  $F(X)$ , and Bob responds with  $F(s)$ , where  $s \in X \cap Y$  is some intersection query. The original secret key (before our transformation) is the first bit  $s^1$ . After our transformation, the secret key is  $Y_{\ell+1}^1$ , and as part of  $M_2$ , Bob sends Alice the bit  $b = s^1 \oplus Y_{\ell+1}^1$  so that she can extract  $Y_{\ell+1}^1$ .

From Alice’s perspective, given  $X, F(X)$ , Bob’s message  $M_2 = F(s)$ ,  $b$  fixes  $Y_{\ell+1}^1$  to the value  $b \oplus s^1$ . (We ignore here the tiny probability that  $s$  cannot be uniquely computed from  $X, F(X)$  and  $F(s)$ , i.e., the probability of a collision in  $F$ .) However, from Eve’s perspective, because she does not know  $X, F(X)$  and she asks no queries (there are no heavy queries in Merkle’s puzzles), the intersection element  $s$  remains uniformly random. When Eve samples  $Y_{\ell+1}^1$  given  $M_1, M_2$  and her non-existent heavy queries, the result is random, and completely independent from the true secret key.

We need to show that when the players’ advantage is small, then the example above cannot happen, and Eve’s key agrees with the players’ w.h.p. To this end, we are interested in the difference between Eve’s “pretend distribution”, and the true distribution that the players use to produce the key: if the two distributions are close, then Eve’s chances of guessing the right secret key are roughly the same as Bob’s. The *only difference* between these two distributions is that given  $M_1, M_2$  and  $\mathcal{E}_1, F(\mathcal{E}_1)$  (which the players do not use),

## 40:14 The Communication Complexity of Key-Agreement

- The players' keys are produced according to the *joint distribution*  $(\text{out}^A, \text{out}^B)$ , and in particular, both players have the same answers  $F(S_2)$  to the non-heavy intersection queries  $S_2 = (X \cap Y) \setminus \mathcal{E}_1$ .
- Eve's pretense that she is Bob is carried out *independently* from Alice's view: Eve cannot use the true intersection queries (which she does not know), only what she has learned about them from  $M_1, M_2, V_E^i$ . The joint distribution of Alice and Eve's keys is therefore given by the product distribution  $\text{out}^A \times \text{out}^B$ .

So, we would like to bound the difference between the joint distribution and the product distribution, i.e.,

$$I_f(\text{out}^A; \text{out}^B | M_1, M_2, \mathcal{E}_1, F(\mathcal{E}_1)).$$

Given the conditioning, Alice's output  $\text{out}^A$  is a function of her view,  $X, F(X)$ . Also, we assumed that Bob's output is a function of his queries  $Y$ . Therefore,

$$I_f(\text{out}^A; \text{out}^B | M_1, M_2, \mathcal{E}_1, F(\mathcal{E}_1)) \leq I_f(X, F(X); Y | M_1, M_2, \mathcal{E}_1, F(\mathcal{E}_1)). \quad (2)$$

Now we need to show that given Eve's view, the dependence between  $X, F(X)$  and  $Y$  is bounded in terms of the advantage:

$$I_f(X, F(X); Y | M_1, M_2, \mathcal{E}_1, F(\mathcal{E}_1)) \leq I_f(S_2, F(S_2); M_2 | M_1, Y, \mathcal{E}_1 \cap Y, F(\mathcal{E}_1 \cap Y)). \quad (3)$$

This proof is somewhat tedious; it relies on the fact that  $M_2$  is a function of  $M_1, Y$  and  $F(Y)$ , and on the fact that  $X, F(X)$  are independent of  $Y, F(Y)$  given the intersection queries and answers,  $S_2, F(S_2)$  and  $\mathcal{E}_1, F(\mathcal{E}_1)$ . Intuitively, all the dependence between  $X, F(X)$  and  $Y$  "flows through" what the players learn about the intersection, and the proof of (3) formalizes this intuition.

### Step I: Bounding the advantage after the first round

For the first round, we analyze the players' advantage in terms of KL-divergence, and bound

$$I(S_1, F(S_1); M_1 | X).$$

Notice that we do not use Eve at this point, because we eliminated any a-priori heavy queries, so there is nothing Eve needs to query in order to "understand"  $M_1$ . For the same reason,  $S_1 = X \cap Y$  (there are no heavy queries to remove from the intersection).

We claim that

$$I(S_1, F(S_1); M_1 | X) \leq \delta |M_1|. \quad (4)$$

This is not hard to see: suppose  $X = x$ . Because we got rid of the a-priori heavy queries, every individual query  $q \in x$  has probability at most  $\delta$  of being asked by Bob (otherwise,  $q$  would be heavy). Therefore, for every  $q \in x$ , we have  $\Pr[q \in S_1 | X = x] \leq \delta$ . Because  $M_1$  is generated by Alice without knowing  $S_1$ , and every query is in  $S_1$  only w.p. at most  $\delta$ , intuitively, the information in  $M_1$  "only applies" to the queries in  $S_1$  with probability  $\delta$ . Therefore the information that  $M_1$  gives about  $S, F(S_1)$  is at most  $\delta |M_1|$ .

The actual proof involves a Shearer-like argument for mutual information, similar to the ones used in [6, 15].

**Step II: Bounding the advantage after the second round**

Now we must bound the advantage the players gain after the second round, and show that

$$I_{SD}(S_2, F(S_2); M_2|Y, M_1, \mathcal{E}_1 \cap Y, F(\mathcal{E}_1 \cap Y)) = O(\sqrt{|\mathcal{M}_1| + |\mathcal{M}_2|}). \quad (5)$$

As we said, we switch here to using statistical distance, and we will see why below.

Following the first round, we know that not much is known about the intersection, because Alice's message  $M_1$  did not convey a lot of information about it. So, our proof here proceeds in two steps: first, we "pretend" that *nothing* is known about the intersection, and consider the distribution  $\mu'$  where given  $M_1$  the distribution of  $Y, F(Y)$  is completely independent from  $X$ . We show that under  $\mu'$ , Bob's message  $M_2$  would only convey  $\delta|\mathcal{M}_2|$  bits of information about the intersection. This is very similar to the analysis of the first round, and it is also carried out using KL-divergence. Formally, we show that for the distribution  $\mu'$  where  $Y, F(Y)$  are drawn independently of  $X$ , we have

$$I^{\mu'}(S_2, F(S_2); M_2|Y, M_1, \mathcal{E}_1 \cap Y, F(\mathcal{E}_1 \cap Y)) \leq \delta|\mathcal{M}_2|. \quad (6)$$

The proof relies on the fact that we excluded heavy queries from  $S_2$  (recall that  $S_2 = (X \cap Y) \setminus \mathcal{E}_1$ ), so given the conditioning, any query in  $Y$  can only belong to  $S_2$  with probability at most  $\delta$ .

However,  $\mu'$  is not the real distribution: given  $M_1$ , we do know a little about the intersection, so  $Y, F(Y)$  are not completely independent from  $X$ . Our next step is to switch to statistical distance, and show that the real distribution  $\mu$  (where  $X, Y$  are not independent) and  $\mu'$  (where they are) are close to each other. Therefore, what we showed for  $\mu'$  is also true for  $\mu$ , with the addition of a small penalty corresponding to the distance between  $\mu$  and  $\mu'$ .

Formally, we prove that

$$\begin{aligned} & I_{SD}^{\mu}(S_2, F(S_2); M_2|Y, M_1, \mathcal{E}_1 \cap Y, F(\mathcal{E}_1 \cap Y)) \\ & \leq O\left(I_{SD}^{\mu'}(S_2, F(S_2); M_2|Y, M_1, \mathcal{E}_1 \cap Y, F(\mathcal{E}_1 \cap Y)) + D_{SD}(\mu' \parallel \mu)\right) \end{aligned} \quad (7)$$

Under  $\mu'$ , by (6) and Pinsker's inequality, we have:

$$I_{SD}^{\mu'}(S_2, F(S_2); M_2|Y, M_1, \mathcal{E}_1 \cap Y, F(\mathcal{E}_1 \cap Y)) \leq \sqrt{\delta|\mathcal{M}_2|}. \quad (8)$$

So, under  $\mu'$  the expected amount of information revealed is small.

Next, we bound the difference between  $\mu$  and  $\mu'$ . We show that:

$$I(Y, F(Y); X|M_1) \leq I(S_1, F(S_1); M_1|X).$$

This is quite similar to the proof of Step III above – here we do use standard mutual information, so the proof uses the chain rule, just as we did above. Since we have shown in Step I that  $I(S_1, F(S_1); M_1|X) \leq \delta|\mathcal{M}_1|$ , we conclude using Pinsker's inequality that

$$D_{SD}(\mu' \parallel \mu) \leq \sqrt{D_{KL}(\mu' \parallel \mu)} \leq \sqrt{\delta|\mathcal{M}_1|}. \quad (9)$$

Together, (8) and (9) are the ingredients we need to apply (7), and obtain:

$$\begin{aligned} & I_{SD}^{\mu}(S_2, F(S_2); M_2|Y, M_1, \mathcal{E}_1 \cap Y, F(\mathcal{E}_1 \cap Y)) \\ & \leq O\left(I_{SD}^{\mu'}(S_2, F(S_2); M_2|Y, M_1, \mathcal{E}_1 \cap Y, F(\mathcal{E}_1 \cap Y)) + D_{SD}(\mu' \parallel \mu)\right) \\ & \leq O(\sqrt{\delta(|\mathcal{M}_1| + |\mathcal{M}_2|)}). \end{aligned}$$

---

**References**

---

- 1 Miklós Ajtai and Cynthia Dwork. A public-key cryptosystem with worst-case/average-case equivalence. In *Proceedings of the twenty-ninth annual ACM symposium on Theory of computing*, pages 284–293. ACM, 1997.
- 2 B. Barak and M. Mahmoody. Merkle Puzzles Are Optimal - An  $O(n^2)$ -Query Attack on Any Key Exchange from a Random Oracle. In *Advances in Cryptology – CRYPTO '09*, pages 374–390, 2009.
- 3 Daniel J Bernstein and Tanja Lange. eBACS: ECRYPT benchmarking of cryptographic systems. <https://bench.cr.yp.to>, accessed 15 May 2018. URL: <https://bench.cr.yp.to>.
- 4 Benny Chor and Eyal Kushilevitz. A zero-one law for boolean privacy. *SIAM Journal on Discrete Mathematics*, 4(1):36–47, 1991.
- 5 Whitfield Diffie and Martin E. Hellman. New Directions in Cryptography. *IEEE Transactions on Information Theory*, 22(6):644–654, 1976.
- 6 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. ACM, 2015.
- 7 Iftach Haitner, Jonathan J. Hoch, Omer Reingold, and Gil Segev. Finding Collisions in Interactive Protocols - Tight Lower Bounds on the Round and Communication Complexities of Statistically Hiding Commitments. *SIAM Journal on Computing*, 44(1):193–242, 2015. Preliminary version in *STOC'07*.
- 8 Iftach Haitner, Noam Mazon, Rotem Oshman, Omer Reingold, and Amir Yehudayoff. On the Communication Complexity of Key-Agreement Protocols. In *Electronic Colloquium on Computational Complexity (ECCC)*, volume 25, page 31, 2018.
- 9 Iftach Haitner, Eran Omri, and Hila Zarosim. Limits on the usefulness of random oracles. *Journal of Cryptology*, 29(2):283–335, 2016.
- 10 Russell Impagliazzo and Steven Rudich. Limits on the provable consequences of one-way permutations. In *Proceedings of the 21st Annual ACM Symposium on Theory of Computing (STOC)*, pages 44–61. ACM Press, 1989.
- 11 Mohammad Mahmoody, Hemanta K Maji, and Manoj Prabhakaran. Limits of random oracles in secure computation. *arXiv preprint*, 2012. [arXiv:1205.3554](https://arxiv.org/abs/1205.3554).
- 12 Robert J McEliece. A public-key cryptosystem based on algebraic. *Coding Thv*, 4244:114–116, 1978.
- 13 Ralph C. Merkle. Secure Communications over Insecure Channels. In *SIMMONS: Secure Communications and Asymmetric Cryptosystems*, 1982.
- 14 Michael O Rabin. Digitalized signatures and public-key functions as intractable as factorization. Technical report, Massachusetts Inst of Tech Cambridge Lab for Computer Science, 1979.
- 15 Anup Rao and Makrand Sinha. Simplified Separation of Information and Communication. In *Electronic Colloquium on Computational Complexity (ECCC)*, volume 22 (57), pages 2–3, 2015.
- 16 Alexander A. Razborov. On the distributional complexity of disjointness. *Theoretical Computer Science*, 106(2):385–390, 1992.
- 17 Ronald L. Rivest, Adi Shamir, and Leonard M. Adelman. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, 21(2):120–126, 1978.

# The Paulsen Problem Made Simple

Linus Hamilton<sup>1</sup>

Massachusetts Institute of Technology, 77 Massachusetts Ave, USA  
luh@mit.edu

Ankur Moitra<sup>2</sup>

Massachusetts Institute of Technology, 77 Massachusetts Ave, USA  
moitra@mit.edu

---

## Abstract

The Paulsen problem is a basic problem in operator theory that was resolved in a recent four-decade work of Kwok, Lau, Lee and Ramachandran. In particular, they showed that every  $\epsilon$ -nearly equal norm Parseval frame in  $d$  dimensions is within squared distance  $O(\epsilon d^{13/2})$  of an equal norm Parseval frame. We give a dramatically simpler proof based on the notion of radial isotropic position, and along the way show an improved bound of  $O(\epsilon d^2)$ .

**2012 ACM Subject Classification** Theory of computation → Design and analysis of algorithms

**Keywords and phrases** radial isotropic position, operator scaling, Paulsen problem

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2019.41

**Related Version** <https://arxiv.org/abs/1809.04726>

**Acknowledgements** We thank Avi Wigderson for numerous helpful comments on an earlier draft. We would also like to thank anonymous referees for helpful comments about the exposition.

## 1 Introduction

The Paulsen problem is a basic problem in operator theory that was resolved in a recent work of Kwok, Lau, Lee and Ramachandran [12]. To state the problem, we need the following definition:

► **Definition 1.** We say that a set of vectors  $v_1, v_2, \dots, v_n \in \mathbb{R}^d$  is an *equal norm Parseval frame* if

$$\sum_{i=1}^n v_i v_i^T = I \text{ and } \|v_i\|_2^2 = \frac{d}{n} \text{ for each } i.$$

Alternatively, we say that it is an  $\epsilon$ -nearly equal norm Parseval frame if

$$(1 - \epsilon)I \preceq \sum_{i=1}^n v_i v_i^T \preceq (1 + \epsilon)I \text{ and } (1 - \epsilon)\frac{d}{n} \leq \|v_i\|_2^2 \leq (1 + \epsilon)\frac{d}{n} \text{ for each } i.$$

When we drop the condition on the norm of each vector, we refer to the set of vectors as a *Parseval frame* or an  $\epsilon$ -nearly Parseval frame respectively. Let  $\mathcal{F}$  denote the set of

---

<sup>1</sup> This work was supported in part by a Fannie and John Hertz Foundation Fellowship.

<sup>2</sup> This work was supported in part by NSF CAREER Award CCF-1453261, NSF Large CCF-1565235, a David and Lucile Packard Fellowship and an Alfred P. Sloan Fellowship.



## 41:2 The Paulsen Problem Made Simple

all equal norm Parseval frames. Lastly for two sequences of vectors  $V = v_1, v_2, \dots, v_n$  and  $W = w_1, w_2, \dots, w_n$  of the same length, we let

$$\text{dist}^2(V, W) = \sum_{i=1}^n \|v_i - w_i\|^2.$$

With this terminology in hand, the Paulsen problem asks:

► **Conjecture 2.** *For every  $\epsilon$ -nearly equal norm Parseval frame  $V$ , is*

$$\inf_{W \in \mathcal{F}} \text{dist}^2(V, W)$$

*bounded by a fixed polynomial in  $\epsilon$  and  $d$ ?*

See [12] and references therein for a detailed account of the history of the Paulsen problem along with earlier bounds on the squared distance that were polynomial in  $\epsilon$ ,  $d$  and  $n$ . Through a tour-de-force utilizing operator scaling, connections to dynamical systems and ideas from smoothed analysis, Kwok, Lau, Lee and Ramachandran [12] proved that the squared distance is at most  $O(\epsilon d^{13/2})$ . The paper was 104 pages long and highly complex. Our main result is a dramatically simpler proof of the Paulsen conjecture, that also yields a much better bound:

► **Theorem 3 (Main).** *For any  $\epsilon$ -nearly equal norm Parseval frame  $V$ , there is an equal norm Parseval frame  $W$  with*

$$\text{dist}^2(V, W) \leq 20\epsilon d^2.$$

In terms of lower bounds, Cahill and Casazza [4] gave a family of examples of  $\epsilon$ -nearly equal norm Parseval frames where the squared distance to the closest equal norm Parseval frame is at least  $\Omega(\epsilon d)$ . It is an interesting open question to close this gap.

Our main idea is to make use of the notion of *radial isotropic position*<sup>3</sup>. In the next section, we define it formally. But to understand it informally, it is useful to compare it to the more familiar notion of placing a set of vectors in isotropic position: Given a set of vectors  $V = v_1, v_2, \dots, v_n \in \mathbb{R}^d$ , is there an invertible affine transformation that generates a new set of vectors  $Y = Av_1 + b, Av_2 + b, \dots, Av_n + b$  that has mean zero and identity covariance? It is well known that there is such a transformation if and only if  $\sum_i v_i v_i^T$  has full rank.

However such a transformation can also stretch out some directions much more than others – e.g. if all but one of the vectors in  $V$  are contained in a  $d - 1$ -dimensional subspace. In this case, the set of vectors after applying the transformation would be quite far from where it started out, in total squared distance. Informally, radial isotropic position asks for a linear transformation  $A$  so that the *renormalized* vectors  $w_i = Av_i / \|Av_i\|$  have the property that  $\sum_i w_i w_i^T$  is a scalar multiple of the identity. The transformation is now nonlinear but is particularly well suited for constructing a close by equal norm Parseval frame.

One can now ask the same sort of question as before: When can a set of vectors be placed in radial isotropic position? Barthe [2] gave a complete characterization of when this is and is not possible which in turn plays a key role in our proof. It turns out that a sufficient condition is that every  $d$  vectors are linearly independent. Now we construct an equal norm Parseval frame as follows: First we renormalize the vectors in  $V$  and then we perturb them.

---

<sup>3</sup> This concept goes by many other names in the literature, such as *well-spread vectors* [5] or *geometric scaling for rank one Brascamp-Lieb datum* [9]. The name we use here originated in [11].

Perturbations play a delicate role in [12]. They give a dynamical system which constructs an equal norm Parseval frame from an  $\epsilon$ -nearly equal norm Parseval frame as its input. In order to bound the total squared distance between the input and output, they need to lower bound the convergence rate. They do this through a certain pseudorandom property (Definition 4.3.2) which they show holds when the input is appropriately perturbed. In our proof, all we need is that the perturbations do not move the set of points by too much in squared distance and that afterwards every  $d$  of them are linearly independent<sup>4</sup>. The latter condition guarantees that there is a linear transformation that places them in radial isotropic position. Let  $W$  be the set of vectors, after applying the linear transformation and renormalizing. By definition, it is an equal norm Parseval frame. Our main technical contribution is in bounding the squared distance between  $V$  and  $W$ , which we do through some elementary but subtle algebraic manipulations.

Taking a step back, the notion of radial isotropic position seems quite powerful and mysterious but has thus far only found a handful of applications. Forster [7] used it to prove a remarkable lower bound in communication complexity (by lower bounding the sign rank of the Hadamard matrix). Hardt and Moitra [11] gave the first algorithm for computing the transformation that places a set of vectors in radial isotropic position (under a slight strengthening of Barthe's conditions). They also gave applications to linear regression in the presence of outliers. Dvir, Saraf and Wigderson [5] used it to prove superquadratic lower bounds for 3-query locally correctable codes over the reals. Here we use it to give a simple proof of the Paulsen conjecture. Are there other exciting applications waiting to be discovered?

## Connections to Operator Scaling and the Brascamp-Lieb Inequality

Radial isotropic position is itself a special case of the more general notion of *geometric position* [1, 2] where we are given an  $n$  tuple of linear transformations  $B_1, B_2, \dots, B_n$  of dimensions  $d_1 \times d, d_2 \times d, \dots, d_n \times d$  and a nonnegative vector  $c$  of dimension  $n$  with  $\sum_{i=1}^n c_i d_i = d$  and the goal is to find square, invertible matrices  $A_1, A_2, \dots, A_n$  and  $A$  so that

$$\sum_{i=1}^n c_i (A_i^{-1} B_i A)^T (A_i^{-1} B_i A) = I \text{ and } (A_i^{-1} B_i A) (A_i^{-1} B_i A)^T = I \text{ for each } i.$$

If we set  $d_i = 1$  for all  $i$ , then each linear transformation  $B_i$  can be written as the inner-product with some vector  $v_i$ . Now if we also set  $c_i = \frac{d}{n}$  for all  $i$ , it is easy to check that  $A$  places the set of vectors  $v_1, v_2, \dots, v_n$  in radial isotropic position.

It turns out that having  $A_1, A_2, \dots, A_n$  and  $A$  that place  $B_1, B_2, \dots, B_n$  in geometric position with respect to the vector  $c$  yields an explicit expression for the best constant  $C$  for which the inequality

$$\int_{x \in \mathbb{R}^d} \prod_{i=1}^n (f_i(B_i x))^{c_i} dx \leq C \prod_{i=1}^n \left( \int_{x_i \in \mathbb{R}^{d_i}} f_i(x_i) dx_i \right)^{c_i}$$

holds over all  $m$  tuples of nonnegative functions  $f_1, f_2, \dots, f_m$  [3]. This is called the Brascamp-Lieb inequality.

<sup>4</sup> In particular, essentially all sufficiently small perturbations would work for us. It could even be an infinitesimal perturbation because we do not need any quantitative bounds on how far they are from having a non-trivial linear dependence.



Finally, in terms of how to compute  $A_1, A_2, \dots, A_n$  and  $A$ , a popular approach is *operator scaling* [10] and there has been considerable recent progress in bounding the number of iterations it needs [8, 9]. As we mentioned, Kwok, Lau, Lee and Ramachandran [12] used operator scaling to solve the Paulsen conjecture. In this sense, our approach and theirs are closely related in that they both revolve around algorithms (in our case the ellipsoid algorithm) for computing radial isotropic position. Perhaps the main technical divergence is that they track how the squared distance changes after each iteration of operator scaling, while we are able to bound the squared distance just based on transformation that places  $v_1, v_2, \dots, v_n$  into radial isotropic position. It is also worth mentioning that if instead of proving existence of a nearby equal norm Parseval frame, we want to find it up to some target precision  $\delta$ , the approaches based on operator scaling typically require the number of iterations to be polynomial in  $1/\delta$ . In contrast, we will give algorithms whose running time is polynomial in  $\log 1/\delta$ .

## 2 Radial Isotropic Position and the Proof

First we introduce some of the basic concepts and results about radial isotropic position. We will do so in slightly more generality than we will ultimately need.

► **Definition 4.** We say that a set of vectors  $u_1, u_2, \dots, u_n \in \mathbb{R}^d$  is in *radial isotropic position* with respect to a coefficient vector  $c \in \mathbb{R}^n$  if

$$\sum_{i=1}^n c_i \left( \frac{u_i}{\|u_i\|} \right) \left( \frac{u_i}{\|u_i\|} \right)^T = I.$$

Note that if we take the trace of both sides in the expression, we get the necessary condition that  $\sum_{i=1}^n c_i = d$ . In fact we will only ever consider the case when each  $c_i = \frac{d}{n}$ . We will also need the following key definition:

► **Definition 5** ([6]). For a set  $U$  of vectors  $u_1, u_2, \dots, u_n \in \mathbb{R}^d$ , its *basis polytope* is defined as

$$\mathcal{B}(U) = \left\{ c \in \mathbb{R}^n \text{ s.t. } \sum_{i=1}^n c_i = d \text{ and for all } A \subseteq [n], \dim(\text{span}\{u_i\}_{i \in A}) \geq \sum_{i \in A} c_i \right\}.$$

Now we are ready to state Barthe's theorem:

► **Theorem 6** ([2]). *A set of vectors  $U = u_1, u_2, \dots, u_n \in \mathbb{R}^d$  can be put into radial isotropic position with respect to  $c$  by a linear transformation if and only if  $c \in \mathcal{B}(U)$ .*

Some further remarks: (1) The usual definition of the basis polytope is based on taking the convex hull of the indicators of subsets of vectors in  $U$  that form a basis. (2) The alternative definition we gave in Definition 5 will be more directly useful for our purposes, and was proven to be equivalent by Edmonds [6]. He used this equivalence to give a separation oracle for the basis polytope, which in turn plays a key role in the algorithm of Hardt and Moitra [11] for computing the linear transformation that puts a set of vectors into radial isotropic position.

See <https://arxiv.org/abs/1809.04726> for the proof of the main theorem. The main idea is to renormalize and then perturb the vectors in  $V$ . After the perturbation, we can invoke Theorem 6 to find a transformation  $A$  that places the vectors in radial isotropic position. We show that we can assume without loss of generality that  $A$  is a nonnegative diagonal matrix whose entries are sorted in non-increasing order along the diagonal. Our main technical lemma gives a bound on the squared distance:

► **Lemma 7.** *Suppose  $U$  is a  $4\epsilon$ -nearly Parseval frame and  $A$  is an entrywise diagonal matrix that puts  $U$  in radial isotropic position. Also suppose that for each  $i$*

$$(1 - \gamma') \frac{d}{n} \leq \|u_i\|_2^2 \leq (1 + \gamma') \frac{d}{n}.$$

Now set

$$W = w_1, w_2, \dots, w_n \text{ with } w_i \triangleq \sqrt{\frac{d}{n}} \left( \frac{Au_i}{\|Au_i\|} \right).$$

Then we have that  $\text{dist}^2(U, W) \leq 8\epsilon d^2 + 4\gamma' d^2$ .

See <https://arxiv.org/abs/1809.04726> for the proof of the main technical lemma.

### 3 An Algorithm for the Paulsen Problem

Every step of the proof of Theorem 3 is straightforward to implement algorithmically, except for the step where we compute the transformation  $A$  that places the set of vectors  $U$  in radial isotropic position. Fortunately, Hardt and Moitra [11] gave an algorithm for computing  $A$  under a slight strengthening of Barthe's conditions which holds in our setting. Informally, they require the vector  $c$  to be strictly inside the basis polytope according to the following notion of scaling:

► **Definition 8.** Let  $(1 - \alpha)\mathcal{B}(U)$  denote the set of vectors  $c$  with the following properties: (1)  $\sum_{i=1}^n c_i = d$ , (2)  $0 \leq c_i \leq 1$  for all  $i$  and (3) for all nonnegative directions  $u$  with  $u_{\min} = 0$ ,

$$(1 - \alpha) \max_{v \in \mathcal{B}(U)} u^T v \geq u^T c.$$

We will state a special case of their main theorem, which is sufficient for our purposes.

► **Theorem 9** ([11]). *Let  $\delta > 0$  and  $\alpha > 0$ . Suppose  $U = u_1, u_2, \dots, u_n \in \mathbb{R}^d$  has the property that every set of  $d$  vectors are linearly independent. Then given  $c \in (1 - \alpha)\mathcal{B}(U)$ , there is an algorithm to find a linear transformation  $A$  so that*

$$\sum_{i=1}^n c_i \left( \frac{Au_i}{\|Au_i\|} \right) \left( \frac{Au_i}{\|Au_i\|} \right)^T = I + J$$

where  $\|J\|_\infty \leq \delta$  – i.e. the largest entry of  $J$  in absolute value is at most  $\delta$ . The running time is polynomial in  $1/\alpha$ ,  $\log 1/\delta$  and  $L$  where  $L$  is an upper bound on the bit complexity of  $U$  and  $c$ .

By combining their algorithm with our proof of Theorem 3 we get:

► **Corollary 10.** *Suppose  $V = v_1, v_2, \dots, v_n \in \mathbb{R}^d$  is an  $\epsilon$ -nearly equal norm Parseval frame. Furthermore suppose  $n > d$ . Then given  $\delta > 0$ , there is an algorithm to compute a  $\delta$ -nearly equal norm Parseval frame  $W$  with*

$$\text{dist}^2(V, W) \leq 20\epsilon d^2$$

whose running time is polynomial in  $\log 1/\delta$  and  $L$  where  $L$  is an upper bound on the bit complexity of  $V$ .

See <https://arxiv.org/abs/1809.04726> for the proof of the corollary. This answers an open question of [12], where they ask whether there is an algorithm for finding an equal norm Parseval frame up to some precision  $\delta$  whose running time is polynomial in  $\log 1/\delta$ .

---

**References**

---

- 1 Keith Ball. Volumes of sections of cubes and related problems. In *Geometric aspects of functional analysis*, pages 251–260. Springer, 1989.
- 2 Franck Barthe. On a reverse form of the Brascamp-Lieb inequality. *Inventiones mathematicae*, 134(2):335–361, 1998.
- 3 Jonathan Bennett, Anthony Carbery, Michael Christ, and Terence Tao. The Brascamp-Lieb inequalities: finiteness, structure and extremals. *Geometric and Functional Analysis*, 17(5):1343–1415, 2008.
- 4 Jameson Cahill and Peter G Casazza. The Paulsen problem in operator theory. *submitted to Operators and Matrices*, 2011.
- 5 Zeev Dvir, Shubhangi Saraf, and Avi Wigderson. Superquadratic Lower Bound for 3-Query Locally Correctable Codes over the Reals. *Theory of Computing*, 13(1):1–36, 2017.
- 6 Jack Edmonds. Submodular functions, matroids, and certain polyhedra. *Combinatorial structures and their applications*, pages 69–87, 1970.
- 7 Jürgen Forster. A linear lower bound on the unbounded error probabilistic communication complexity. *Journal of Computer and System Sciences*, 65(4):612–625, 2002.
- 8 Ankit Garg, Leonid Gurvits, Rafael Oliveira, and Avi Wigderson. A deterministic polynomial time algorithm for non-commutative rational identity testing. In *Foundations of Computer Science (FOCS), 2016 IEEE 57th Annual Symposium on*, pages 109–117. IEEE, 2016.
- 9 Ankit Garg, Leonid Gurvits, Rafael Oliveira, and Avi Wigderson. Algorithmic and optimization aspects of Brascamp-Lieb inequalities, via Operator Scaling. *Geometric and Functional Analysis*, 28(1):100–145, 2018.
- 10 Leonid Gurvits. Classical complexity and quantum entanglement. *Journal of Computer and System Sciences*, 69(3):448–484, 2004.
- 11 Moritz Hardt and Ankur Moitra. Algorithms and hardness for robust subspace recovery. In *Conference on Learning Theory*, pages 354–375, 2013.
- 12 Tsz Chiu Kwok, Lap Chi Lau, Yin Tat Lee, and Akshay Ramachandran. The Paulsen problem, continuous operator scaling, and smoothed analysis. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing*, pages 182–189. ACM, 2018.

# How to Subvert Backdoored Encryption: Security Against Adversaries that Decrypt All Ciphertexts

Thibaut Horel<sup>1</sup>

Harvard University, Cambridge, MA, USA

Sunoo Park<sup>2</sup>

MIT, Cambridge, MA, USA

Silas Richelson

University of California, Riverside, CA, USA

Vinod Vaikuntanathan<sup>3</sup>

MIT, Cambridge, MA, USA

---

## Abstract

In this work, we examine the feasibility of secure and undetectable point-to-point communication when an adversary (e.g., a government) can read all encrypted communications of surveillance targets. We consider a model where the only permitted method of communication is via a government-mandated encryption scheme, instantiated with government-mandated keys. Parties cannot simply encrypt ciphertexts of some other encryption scheme, because citizens caught trying to communicate outside the government's knowledge (e.g., by encrypting strings which do not appear to be natural language plaintexts) will be arrested. The one guarantee we suppose is that the government mandates an encryption scheme which *is* semantically secure against outsiders: a perhaps reasonable supposition when a government might consider it advantageous to secure its people's communication against foreign entities. But then, what good is semantic security against an adversary that holds all the keys and has the power to decrypt?

We show that even in the pessimistic scenario described, citizens *can* communicate securely and undetectably. In our terminology, this translates to a positive statement: all semantically secure encryption schemes support *subliminal communication*. Informally, this means that there is a two-party protocol between Alice and Bob where the parties exchange ciphertexts of what appears to be a normal conversation even to someone who knows the secret keys and thus can read the corresponding plaintexts. And yet, at the end of the protocol, Alice will have transmitted her secret message to Bob. Our security definition requires that the adversary not be able to tell whether Alice and Bob are just having a normal conversation using the mandated encryption scheme, or they are using the mandated encryption scheme for subliminal communication.

Our topics may be thought to fall broadly within the realm of *steganography*. However, we deal with the non-standard setting of an adversarially chosen distribution of cover objects (i.e., a stronger-than-usual adversary), and we take advantage of the fact that our cover objects are ciphertexts of a semantically secure encryption scheme to bypass impossibility results which we show for broader classes of steganographic schemes. We give several constructions of subliminal communication schemes under the assumption that key exchange protocols with pseudorandom messages exist (such as Diffie-Hellman, which in fact has truly random messages).

---

<sup>1</sup> Supported, in part, by the National Science Foundation under grants CAREER IIS-1149662, and CNS-1237235, by the Office of Naval Research under grants YIP N00014-14-1-0485 and N00014-17-1-2131, and by a Google Research Award.

<sup>2</sup> Supported by the Center for Science of Information STC (CSoI), an NSF Science and Technology Center (grant agreement CCF-0939370), MACS project NSF grant CNS-1413920, and a Simons Investigator Award Agreement dated 2012-06-05.

<sup>3</sup> Supported in part by NSF Grants CNS-1350619, CNS-1414119 and CNS-1718161, Alfred P. Sloan Research Fellowship, Microsoft Faculty Fellowship and a Steven and Renee Finn Career Development Chair from MIT.



## 42:2 How to Subvert Backdoored Encryption

**2012 ACM Subject Classification** Theory of computation → Cryptographic protocols

**Keywords and phrases** Backdoored Encryption, Steganography

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2019.42

**Related Version** A full version of the paper is available at [17], <https://eprint.iacr.org/2018/212>.

**Acknowledgements** We are grateful to Omer Paneth and Adam Sealfon for insightful remarks on an early draft.

### 1 Introduction

Suppose that we lived in a world where the government wished to read all the communications of its citizens, and thus decreed that citizens must not communicate in any way other than by using a specific, government-mandated encryption scheme with government-mandated keys. Even face-to-face communication is not allowed: in this Orwellian world, anyone who is caught speaking to another person will be arrested for treason. Similarly, anyone whose communications appear to be hiding information will be arrested: e.g., if the plaintexts encrypted using the government-mandated scheme are themselves ciphertexts of a different encryption scheme. However, the one assumption that we entertain in this paper, is that the government-mandated encryption scheme is, in fact, semantically secure: this is a tenable supposition with respect to a government that considers secure encryption to be in its interest, in order to prevent foreign powers from spying on its citizens' communications.

A natural question then arises: is there any way that the citizens would be able to communicate in a fashion undetectable to the government, based only on the semantic security of the government-mandated encryption scheme, and *despite the fact that the government knows the keys and has the ability to decrypt all ciphertexts*?<sup>4</sup> What can semantic security possibly guarantee in a setting where the adversary has the private keys?

This question may appear to fall broadly within the realm of *steganography*: the science of hiding secret communications within other innocent-looking communications (called “cover objects”), in an undetectable way. Indeed, it can be shown that if two parties have a shared secret, then based on slight variants of existing techniques for *secret-key steganography*, they can conduct communications hidden from the government.<sup>5</sup>

However, the question of whether two parties who have never met before can conduct hidden communications is more interesting. This is related to the questions of *public-key steganography* and *steganographic key exchange* which were both first formalized by von Ahn and Hopper [23]. Public-key steganography is inadequate in our setting since exchanging or publishing public keys is potentially conspicuous and thus is not an option in our setting. All prior constructions of steganographic key exchange require the initial sampling of a public random string that serves as a public parameter of the steganographic scheme. Intuitively, in these constructions, the public random string can be thought to serve the purpose of

---

<sup>4</sup> We note that one could, alternatively, consider an adversary with decryption capabilities arising from possession of some sort of “backdoor.” For the purposes of this paper, we opted for the simpler and still sufficiently expressive model where the adversary’s decryption power comes from knowledge of all the decryption keys.

<sup>5</sup> We refer the reader to Section 1.4 for more details.

selecting a specific steganographic scheme from a family of schemes *after* the adversary has chosen a strategy. That is, the schemes crucially assume that the adversary (the dystopian government, in our story above) cannot choose its covert distribution as a function of the public parameter.

It is conservative and realistic to expect a malicious adversary to choose the covert distribution *after* the honest parties have decided on their communication protocol (including the public parameters). After all, malice never sleeps [18]. Alas, we show that if the covert distribution is allowed to depend on the communication protocol, steganographic communication is impossible. In other words, for every purported steganographic communication protocol, there is a covert distribution (even one with high min-entropy) relative to which the communication protocol fails to embed subliminal messages. The relatively simple counterexample we construct is inspired by the impossibility of deterministic extraction.

**Semantic Security to the Rescue?** However, this impossibility result does not directly apply to our setting, as our covert distribution is restricted to be a sequence of ciphertexts (that may encrypt arbitrary messages). Moreover, the ciphertexts are semantically secure against entities that are not privy to the private keys. We define the notion of a *subliminal communication scheme* (Definition 7) as a steganographic communication scheme where security holds relative to covert distributions that are guaranteed to be ciphertexts of some semantically secure encryption scheme. Is there a way to use semantic security to enable subliminal communication?

Our first answer to this question is negative. In particular, consider the following natural construction: first, design an extractor function  $f$ ; then, to subliminally transmit a message bit  $b$ , sample encryptions  $c$  of a (even adversarially prescribed) plaintext  $m$  using independent randomness every time, until  $f(c) = b$ . There are two reasons this idea does not work. First, if the plaintext bit  $b$  is not random, the adversary can detect this by applying the extractor function  $f$  to the transmitted covert. Second, the government can pick an adversarial (semantically secure) encryption scheme where the extractor function  $f$  is constant on all ciphertexts; this is again similar to the impossibility of deterministic extraction.

Nevertheless, we show how to circumvent these difficulties and use the semantic security of the underlying (adversarial) encryption scheme to construct subliminal communication.

► **Theorem 1** (Informal version of Theorem 11). *Under the decisional Diffie-Hellman (DDH) assumption – or any other assumption that gives rise to a key exchange protocol with messages indistinguishable from random – there is a subliminal communication scheme which allows the transmission of  $O(\log \kappa)$  many bits per ciphertext after a setup phase of  $\tilde{O}(\log \kappa)$  ciphertexts ( $\kappa$  is the security parameter).*

We then show how to improve our first construction to reduce the length of the setup phase under additional assumptions.

## 1.1 Overview of Our Construction

The first idea in our construction is implicit in essentially all the works in steganography starting from [20]: namely, to achieve subliminal communication of arbitrary messages, it is sufficient to be able to undetectably communicate *uniformly randomly distributed strings of one's choice*. In other words, Alice samples a string  $r$  which is randomly distributed, produces some ciphertext(s) to be sent to Bob, such that Bob is able to learn  $r$  from them, and yet a PPT eavesdropper Eve who sees the entire communication transcript cannot distinguish between the following two cases:

1. Alice is indeed sending (hereafter, “embedding”) random strings to Bob, or
2. Alice is producing ciphertexts using the unmodified government-mandated encryption algorithm, without embedding such random strings.

To be more precise, the indistinguishability requirement holds for any given (adversarially specified) distribution  $\mathcal{M}$  of message sequences that Alice may choose to encrypt using the government-mandated encryption scheme. Notice that this does not preclude that Eve may be able to learn  $r$  and indeed, our constructions do allow an eavesdropper to learn the embedded strings. Given the ability to undetectably communicate randomly distributed strings, Alice and Bob can then embed to each other the messages of a key-exchange protocol with randomly distributed messages (such as Diffie-Hellman) to establish a shared secret, and then embed to each other ciphertexts of a secret-key encryption scheme with pseudorandom ciphertexts, using the established secret as the key.

All known constructions of such *undetectable random string embedding* rely on the sampling of a public random seed after the adversarial strategy is fixed. In this paper, however, we are interested in bootstrapping hidden communications from the very ground up: we are not willing to assume that the parties start from a state where a seed is already present.

We observe that the ability to embed randomly distributed strings *of one’s choice* – rather than, e.g., to apply a deterministic function to ciphertexts of the government-mandated encryption scheme, and thereby obtain randomly distributed strings which the creator of the ciphertexts did not choose – is crucial to the above-outlined scheme. The notion of undetectably embedding *exogenous* random strings – i.e., strings that are randomly distributed outside of Alice’s control, but both Alice and Bob can read them – is seemingly much weaker, and certainly cannot be used to embed key exchange messages or secret-key ciphertexts. However, we observe that this weaker primitive turns out to be achievable, for our specific setting, without the troublesome starting assumption of a public random seed. We identify a method for embedding *exogenous* random strings into ciphertexts of an adversarially chosen encryption scheme (interestingly, our method does not generalize to embedding into arbitrary min-entropy distributions). We then exploit this method to allow the communicating parties to establish a random seed – from which point they can proceed to embed random strings *of their choice*, as described above.

In building this weaker primitive, in order to bypass our earlier-described impossibility result, we extract from two ciphertexts at a time, instead of one. We begin with the following simple idea: for each consecutive pair of ciphertexts  $c$  and  $c'$ , a single hidden (random) bit  $b$  is defined by  $b = f(c, c')$  where  $f$  is some two-source extractor. It is initially unclear why this should work because (1)  $c$  and  $c'$  are encryptions of messages  $m$  and  $m'$  which are potentially dependent, and two-source extractors are not guaranteed to work without independence; and (2) even if this difficulty could be overcome, ciphertexts of semantically secure encryption scheme can have min-entropy as small as  $\omega(\log \kappa)$  (where  $\kappa$  is the security parameter) and no known two-source extractor known can extract from such a small min-entropy.

We overcome difficulty (1) by relying on the semantic security of the ciphertexts of the adversarially chosen encryption scheme. Paradoxically, even though the adversary knows the decryption key, we exploit the fact that semantic security still holds against the *extractor*, which does not have the decryption key. The inputs in our case are ciphertexts which are not necessarily independent, but semantic security implies that they are computationally indistinguishable from being independent. Thus, the output of  $f(c, c')$  is pseudorandom. Indeed, when  $f$  outputs a single bit (as in our construction), the output is also statistically close to random. The crucial point here is that the semantic security of the encryption scheme is used not against the government, but rather against the extraction function  $f$ .



Our next observation, to address difficulty (2), is that the ciphertexts are not only computationally independent, but they are also computationally indistinguishable from i.i.d. In particular, each pair of ciphertexts is indistinguishable from a pair of encryptions of 0, by semantic security. Based on this observation, we can use a very simple “extractor”, namely, the greater-than function GT. In fact, GT is an extractor with two input sources, whose output bit has negligible bias when the sources have  $\omega(\log \kappa)$  min-entropy and are *independently and identically distributed* (this appears to be a folklore observation; see, e.g., [3]). Because of the last condition, GT is not a true two-source extractor according to standard definitions, but is still suitable for our setting.

By repeatedly extracting random bits from pairs of consecutive ciphertexts using GT, Alice and Bob can construct a shared random string  $s$ . Note that in this process, Alice and Bob generate ciphertexts using the unmodified government-mandated encryption scheme, so the indistinguishability requirement clearly holds. We stress again that  $s$  is also known to a passive eavesdropper of the communication. This part of our construction, up to the construction of the string  $s$ , is presented in details in Section 5.1. From there, constructing a subliminal communication scheme is not hard: Alice and Bob use  $s$  as the seed of a strong seeded extractor to subliminally communicate random strings *of their choice* as explained in Section 5.2. The complete description of our protocol is given in Section 5.3.

## 1.2 Improved Constructions for Specific Cases

While our first construction has the advantage of simplicity, the initial phase to agree on shared random string (using the GT function) transmits only one hidden bit per ciphertext of the government-mandated encryption scheme. A natural question is whether this rate of transmission can be improved. We show that if the government-mandated encryption scheme is *succinct* in the sense that the ciphertext expansion factor is at most 2, then it is possible to improve the rate of transmission in this phase to  $O(\log \kappa)$  hidden bits per ciphertext using an alternative construction based on the extractor from [12]. In other words, our first result showed that if the government-mandated encryption scheme is semantically secure, we can use it to communicate subliminally; the second result shows that if the government-mandated encryption scheme is efficient, that is even better for us, in the sense that it can be used for more efficient subliminal communication.

► **Theorem 2 (Informal).** *If there is a secure key exchange protocol whose message distribution is pseudorandom, then there is a subliminal communication scheme in which a shared seed is established in two exchanges of ciphertexts of a succinct encryption scheme.*

Theorem 1 exploited the specific nature of the cover object distribution in our setting (specifically, that a sequence of encryptions of arbitrary messages is indistinguishable from an i.i.d. sequence of encryptions of zero). Theorem 2 exploits an additional consequence of the semantic security of the government-mandated encryption scheme: if it is succinct, then ciphertexts are computationally indistinguishable from sources of high min-entropy (i.e., they have large HILL-entropy).

It may be possible to use more advanced two-source extractors to work with a larger class of encryption schemes (with larger expansion factors); however, the best known such extractors have an inverse polynomial error rate [8] (whereas our construction’s extractor has negligible error). Consequently, designing a subliminal communication protocol using these extractors seems to require additional ideas, and we leave this as an open problem.

Finally, we show yet another approach in cases where the distribution of “innocent” messages to be encrypted under the government-mandated encryption scheme has a certain amount of conditional min-entropy. For such cases, we construct an alternative scheme

that leverages the semantic security of the encryption scheme in a rather different way: namely, the key fact for this alternative construction is that (in the absence of a decryption key) a ciphertext appears independent of the message it encrypts. In this case, running a two-source extractor on the message and the ciphertext works. The resulting improvement in the efficiency of the scheme is comparable to that of Theorem 2.

► **Theorem 3 (Informal).** *If there is a secure key-exchange protocol whose message distribution is pseudorandom, then there is a subliminal communication scheme for any cover distribution that either*

- *consists of ciphertexts of a semantically secure encryption scheme, if the innocent message distribution  $\mathcal{M}$  has conditional min-entropy rate  $1/2$ , or*
- *consists of ciphertexts of a semantically secure and succinct encryption scheme, if the innocent message distribution  $\mathcal{M}$  has conditional min-entropy  $\omega(\log \kappa)$ .*

*In both cases, the shared seed is established during the setup phase in only two exchanges of ciphertexts.*

Due to space constraints, the results described in this subsection (1.2) are not discussed further herein. They are presented in detail in the full version of this paper [17].

### 1.3 Final Introductory Remarks

**On Our Modeling Assumptions.** Our model considers a relatively powerful adversary that, for example, has the ability to choose the encryption scheme using which all parties must communicate, and to decrypt all such communications. We believe that this can be very realistic in certain scenarios, but it is also important to note the limitations that our model places on the adversary.

The most obvious limitation is that the encryption scheme chosen by the adversary must be semantically secure (against third parties that do not have the ability to decrypt). Another assumption is that citizens are able to run algorithms of their choice on their own computers without, for instance, having every computational step monitored by the government. Moreover, citizens may use encryption randomness of their choice when producing ciphertexts of the government-mandated encryption scheme: in fact, this is a key fact that our construction exploits. Interestingly, secrecy of the encryption randomness from the adversary is irrelevant: after all, the adversary can always choose an encryption scheme where the encryption randomness is recoverable given the decryption key. Despite this, the ability of the encryptor to choose the randomness to input to the encryption algorithm can be exploited – as by our construction – to allow for subliminal communication.

**The Meaning of Semantic Security when the Adversary Can Decrypt.** In an alternate light, our work may be viewed as asking the question: *what guarantee, if any, does semantic security provide against adversary in possession of the decryption key?* Our results find, perhaps surprisingly, that some meaningful guarantee is still provided by semantic security even against an adversary is able to decrypt: more specifically, that *any* communication channel allowing transmission of ciphertexts can be leveraged to allow for undetectable communications between two parties that have never met. From this perspective, our work may be viewed as the latest in a scattered series of recent works that consider what guarantees can be provided by cryptographic primitives that are somehow “compromised” – examples of recent works in this general flavor are cited in Section 1.4 below.

**Concrete Security Parameters.** From a more practical perspective, it may be relevant to consider that the government in our hypothetical Orwellian scenario would be incentivized to opt for an encryption scheme with the least possible security level so as to ensure security against foreign powers. In cases where the government considers itself to have more computational power than foreign adversaries (perhaps by a constant factor), this could create an interesting situation where the security parameter with which the government-mandated scheme must be instantiated is *below* what is necessary to ensure security against the government’s own computational power.

Such a situation could be risky for citizens’ hidden communications: intuitively, our constructions guarantee indistinguishability *against the citizens’ own government* between an “innocent” encrypted conversation and one which is carrying hidden subliminal messages. However, the distinguishing advantage in this indistinguishability game *depends on the security parameter* of the government-mandated encryption scheme. Thus, it could be that the two distributions are far enough apart for the citizens’ own government to distinguish (though not for foreign governments to distinguish). We observe that citizens cognizant of this situation can further reduce the distinguishing advantage beyond that provided by our basic construction, using the standard technique of amplifying the proximity of a distribution (which is far from random) to uniformly random, by taking the XOR of several samples from the far-from-random distribution.

Having outlined this potential concern and solution, the rest of the paper will disregard these issues in the interest of clarity of exposition, and present a purely asymptotic analysis.

**Open Problems.** Our work suggests a number of open problems. A natural one is the extent to which the modeling assumptions that this work makes – such as the ability of honest encryptors to use true randomness for encryption – can be relaxed or removed, while preserving the ability to communicate subliminally. For example, one could imagine yet another alternate universe, in which the hypothetical Orwellian government not only mandates that citizens use the prescribed encryption scheme, but also that their encryption randomness must be derived from a specific government-mandated pseudorandom generator.

The other open problems raised by our work are of a more technical nature and better understood in the context of the specific details of our constructions; for this reason we defer their discussion to Section 6.

## 1.4 Other Related Work

The scientific study of steganography was initiated by Simmons more than thirty years ago [20], and is the earliest mention of the term “subliminal channel” referring to the conveyance of information in a cryptosystem’s output in a way that is different from the intended output,<sup>6</sup> of which we are aware. Subsequent works such as [7, 19, 27] initially explored information-theoretic treatments of steganography, and then Hopper, Langford, and von Ahn [16] gave the first complexity-theoretic (secret-key) treatment almost two decades later. Public-key variants of steganographic notions – namely, public-key steganography and steganographic key exchange – were first defined by [23]. There is very little subsequent literature on public-key steganographic primitives; one notable example is by Backes and Cachin [2], which considers public-key steganography against active attacks (their attack model, which is stronger than that of [23], was also considered in [16] but had never been applied to the public-key setting).

---

<sup>6</sup> This phrasing is loosely borrowed from [26].

The alternative perspective of our work as addressing the question of whether any sort of secret communication can be achieved via transmission of ciphertexts of an adversarially designed cryptosystem alone fits into a scattered series of recent works that consider what guarantees can or cannot be provided by compromised cryptographic primitives. For example, Goldreich [14], and later, Cohen and Klein [10], consider what unpredictability guarantee is achieved by the classic GGM construction [15] when the traditionally secret seed is known; Austrin et al. [1] study whether certain cryptographic primitives can be secure even in the presence of an adversary that has limited ability to tamper with honest parties' randomness; Dodis et al. [13] consider what cryptographic primitives can be built based on backdoored pseudorandom generators; and Bellare, Jaeger, and Kane [4] present attacks that work against any symmetric-key encryption scheme, that completely compromise security by undetectably corrupting the algorithms of the encryption scheme (such attacks might, for example, be feasible if an adversary could generate a bad version of a widely used cryptographic library and install it on his target's computer).

The last work mentioned above, [4], is actually part of the broader field of kleptography, originally introduced by Young and Yung [26, 25, 24]. Broadly speaking, a *kleptographic attack* “uses cryptography against cryptography” [26] – i.e., changes the behavior of a cryptographic system in a fashion undetectable to an honest user with black-box access to the cryptosystem, such that the use of the modified system leaks some secret information (e.g., plaintexts or key material) to the attacker who performed the modification. An example of such an attack might be to modify the key generation algorithm of an encryption scheme such that an adversary in possession of a “back door” can derive the private key from the public key, yet an honest user finds the generated key pairs to be indistinguishable from correctly produced ones. Kleptography has enjoyed renewed research activity since [5] introduced a formal model of a specific type of kleptographic attack called *algorithm substitution attacks* (ASAs), motivated by recent revelations suggesting that intelligence agencies have successfully implemented attacks of this nature at scale. Recently, [6] formalized an equivalence between certain variants of ASA and steganography.

Our setting differs significantly from kleptography in that the encryption algorithms are public and not tampered with (i.e., adhere to a purported specification), and in fact may be *known* to be designed by an adversarial party.

## 2 Preliminaries

Proofs of all propositions, lemmata, and theorems are in the full version of this paper [17] due to space constraints.

**Notation.**  $\kappa$  is the security parameter throughout. PPT means “probabilistic polynomial time.”  $[n]$  denotes the set  $\{1, \dots, n\}$ .  $U_n$  is a uniform variable over  $\{0, 1\}^n$ , independent of every other variable in this paper. We write  $X \sim Y$  to express that  $X$  and  $Y$  are identically distributed. Given two variables  $X$  and  $Y$  over  $\{0, 1\}^k$ , we denote by  $\|X - Y\|_s$  the statistical distance defined by:

$$\|X - Y\|_s = \frac{1}{2} \sum_{x \in \{0, 1\}^k} |\Pr[X = x] - \Pr[Y = x]| .$$

For a random variable  $X$ , we define the min-entropy of  $X$  by  $H_\infty(X) = -\log \max_x \Pr[X = x]$ . The collision probability is  $\text{CP}(X) = \sum_x \Pr[X = x]^2$ .

## 2.1 Encryption and Key Exchange

We assume familiarity with the standard notions of semantically secure public-key and private-key encryption, and key exchange. This subsection defines notation and terminology.

**Public-Key Encryption.** We use the notation  $E = (E.Gen, E.Enc, E.Dec)$  for the public-key encryption scheme mandated by the adversary.

**Secret-key Encryption.** We write  $SKE = (SKE.Gen, SKE.Enc, SKE.Dec)$  to denote a secret-key encryption scheme. We define a *pseudorandom secret-key encryption scheme* to be a secret-key encryption scheme whose ciphertexts are indistinguishable from random. It is a standard result that pseudorandom secret-key encryption schemes can be built from one-way functions.

**Key Exchange.** A *key-exchange protocol*  $\Lambda$  is a two-party protocol executed between two parties  $P_0$  and  $P_1$ , where each party outputs a key at the end of the protocol. The *correctness* guarantee for key-exchange Protocols requires that the two outputted keys be equal with overwhelming probability. The security guarantee for key-exchange protocols requires that  $(T, K) \stackrel{c}{\approx} (T, K_{\S})$ , where  $T$  is a key-exchange protocol transcript,  $K$  is the shared key established in  $T$ , and  $K_{\S}$  is a random unrelated key.

We define a *pseudorandom key-exchange protocol* to be a key-exchange protocol whose transcripts are distributed indistinguishably from random. That is, a pseudorandom key-exchange protocol has the stronger guarantee that  $(T, K) \stackrel{c}{\approx} (U, K_{\S})$  where  $U$  is the uniform distribution over message sequences of the appropriate length, where messages are drawn randomly from the message space of  $\Lambda$ .

The classical protocol of Diffie and Hellman [11] is pseudorandom; in fact, its messages are uniformly random over a cyclic group  $G$ . However, the constructions in this paper assume a key-exchange protocol whose messages are pseudorandom *over bit strings*. In fact, it is possible to transform a key-exchange protocol whose messages are pseudorandom over an arbitrary domain  $G \subseteq \{0, 1\}^{\ell}$  into a key-exchange protocol whose messages are pseudorandom over bit strings. Proposition 4, below, gives an encoding and decoding algorithm to transform uniformly random messages in  $G$  into a sequence of uniformly random messages in  $\{0, 1\}^{\ell}$ . The encoding and decoding algorithms run in polynomial time as long as the density of messages  $\frac{|G|}{2^{\ell}}$  is noticeable (i.e., at least  $\frac{1}{\kappa^c}$  for some  $c \geq 1$ ). This is the case, for example, when the Diffie-Hellman protocol is instantiated with the group of quadratic residues modulo a safe prime (in which case the density of message is constant close to  $\frac{1}{2}$ ).

► **Proposition 4.** *Let  $G$  be a subset of  $\{0, 1\}^{\ell}$  and define  $p = \frac{\kappa 2^{\ell}}{|G|}$ . There is an encoding algorithm  $E : G \rightarrow (\{0, 1\}^{\ell})^p$  and a decoding algorithm  $D : (\{0, 1\}^{\ell})^p \rightarrow G \cup \{\perp\}$  that satisfy the following properties:*

1. Correctness: *for all  $g \in G$ ,  $\Pr [D(E(g)) \neq g]$  is negligible in  $\kappa$ .*
  2. Randomness: *for uniformly random  $g \leftarrow G$ ,  $E(g)$  is uniformly random over  $(\{0, 1\}^{\ell})^p$ .*
- Explicit descriptions of algorithms  $E, D$  are given in the full version of this paper.*

## 2.2 Extractors

Next, we give standard definitions of two-source and seeded extractors.

► **Definition 5.** The family  $2\text{Ext} : \{0, 1\}^n \times \{0, 1\}^{n'} \rightarrow \{0, 1\}^{\ell}$  is a  $(k_1, k_2, \varepsilon)$  *two-source extractor* if for all  $\kappa \in \mathbb{N}$  and for all pairs  $(X, Y)$  of independent random variables over  $\{0, 1\}^{n(\kappa)} \times \{0, 1\}^{n'(\kappa)}$  such that  $H_{\infty}(X) \geq k_1(\kappa)$  and  $H_{\infty}(Y) \geq k_2(\kappa)$ , it holds that

$\|2\text{Ext}_\kappa(X, Y) - U_{\ell(\kappa)}\|_{\mathfrak{s}} \leq \varepsilon(\kappa)$ . We say that  $2\text{Ext}$  is *strong w.r.t. the first input* if it satisfies the stronger property that  $\|(X, 2\text{Ext}_\kappa(X, Y)) - (X, U_{\ell(\kappa)})\|_{\mathfrak{s}} \leq \varepsilon(\kappa)$ . A strong two-source extractor w.r.t. the second input is defined analogously. Finally, we say that  $2\text{Ext}$  is a  $(k, \varepsilon)$  *same-source* extractor if  $n = n'$  and the extractor output is only required to be statistically close to uniform when  $(X, Y)$  is a pair of i.i.d. random variables with  $H_\infty(X) = H_\infty(Y) \geq k(\kappa)$ .

► **Definition 6.** The family  $\text{Ext} : \{0, 1\}^n \times \{0, 1\}^{n'} \rightarrow \{0, 1\}^\ell$  is a  $(k, \varepsilon)$  *seeded extractor* if for all  $\kappa \in \mathbb{N}$  and any random variable  $X$  over  $\{0, 1\}^{n'(\kappa)}$  such that  $H_\infty(X) \geq k(\kappa)$ , it holds that  $\|\text{Ext}_\kappa(U_{n(\kappa)}, X) - U_{\ell(\kappa)}\|_{\mathfrak{s}} \leq \varepsilon(\kappa)$ . We say moreover that  $\text{Ext}$  is *strong* if it satisfies the stronger property that  $\|(U_{n(\kappa)}, \text{Ext}_\kappa(U_{n(\kappa)}, X)) - (U_{n(\kappa)}, U_{\ell(\kappa)})\|_{\mathfrak{s}} \leq \varepsilon(\kappa)$ .

### 3 Subliminal Communication

**Conversation Model.** The protocols we will construct take place over a communication between two parties  $P_0$  and  $P_1$  alternately sending each other ciphertexts of a public-key encryption scheme. *W.l.o.g.*, we assume that  $P_0$  initiates the communication, and that communication occurs over a sequence of *exchange-rounds* each of which comprises two sequential messages: in each exchange-round, one party  $P_b$  sends a message to  $P_{1-b}$  and then  $P_{1-b}$  sends a message to  $P_b$ . Let  $m_{b,i}$  denote the plaintext message sent by  $P_b$  to  $P_{1-b}$  in exchange-round  $i$ , and let  $\mathbf{m}_i = (m_{0,i}, m_{1,i})$  denote the pair of messages exchanged. For  $i \geq 1$ , let us denote by  $\tau_{0,i} = (\mathbf{m}_1, \dots, \mathbf{m}_{i-1})$  and  $\tau_{1,i} = (\mathbf{m}_1, \dots, \mathbf{m}_{i-1}, m_{0,i})$  the plaintext transcripts available to  $P_0$  and  $P_1$  respectively during exchange-round  $i$ , in the case when  $P_0$  sends the first message in exchange-round  $i$ .<sup>7</sup> We define  $\tau_{0,0}$  and  $\tau_{1,0}$  to be empty lists (i.e., empty starting transcripts). (Note that when a notation contains both types of subscripts, we write the subscripts denoting the party and round in **blue** and **red** respectively, to improve readability.)

Recall that our adversary has the power to decrypt all ciphertexts under its chosen public-key encryption scheme  $\mathbb{E}$ . Intuitively, it is therefore important that the plaintext conversation between  $P_0$  and  $P_1$  appears innocuous (and does not, for example, consist of ciphertexts of another encryption scheme). To model this, we assume the existence of a next-message distribution  $\mathcal{M}$ , which outputs a next innocuous message given the transcript of the plaintext conversation so far. This is denoted by  $m_{b,i} \leftarrow \mathcal{M}(\tau_{b,i})$ .

In all the protocols we consider, the symbol  $\mathfrak{s}$  is used to denote internal state kept locally by  $P_0$  and  $P_1$ . It is implicitly assumed that each party's state contains an up-to-date transcript of all messages received during the protocol. Parties may additionally keep other information in their internal state, as a function of the local computations they perform. For  $i \geq 1$ ,  $\mathfrak{s}_{b,i}$  denotes the state of  $P_b$  at the conclusion of exchange-round  $i$ . Initial states  $\mathfrak{s}_{b,0} = \emptyset$  are empty.

We begin with a simpler definition that only syntactically allows for the transmission of a single message (Definition 7). This both serves as a warm-up to the multi-message definition presented next (Definition 8), and will be used in its own right to prove impossibility results.

► **Definition 7.** A *subliminal communication scheme* is a two-party protocol:

$$\Pi^{\mathbb{E}} = (\Pi_{0,1}^{\mathbb{E}}, \Pi_{1,1}^{\mathbb{E}}, \Pi_{0,2}^{\mathbb{E}}, \Pi_{1,2}^{\mathbb{E}}, \dots, \Pi_{0,r}^{\mathbb{E}}, \Pi_{1,r}^{\mathbb{E}}; \Pi_{1,\text{out}}^{\mathbb{E}})$$

<sup>7</sup> If instead  $P_1$  spoke first in round  $i$ , then  $\tau_{0,i}$  would contain  $m_{1,i}$ , and  $\tau_{1,i}$  would not contain  $m_{0,i}$ .



where  $r \in \text{poly}$  is the number of exchange-rounds and each  $\Pi_{b,i}^E$  is a PPT algorithm with oracle access to the algorithms of a public-key encryption scheme  $E$ . Party  $P_0$  is assumed to receive as input a message  $\text{msg}$  (of at least one bit) that is to be conveyed to  $P_1$  in an undetectable fashion. The algorithms  $\Pi_{b,i}^E$  are used by  $P_b$  in round  $i$ , respectively, and  $\Pi_{1,\text{out}}^E$  denotes the algorithm run by  $P_1$  to produce an output  $\text{msg}'$  at the end of the protocol.

A subliminal communication scheme must satisfy the following syntax, correctness and security guarantees.

- **Syntax.** In each exchange-round  $i = 1, \dots, r$ :
  1.  $P_0$  performs the following steps:
    - a. Sample “innocuous message”  $m_{0,i} \leftarrow \mathcal{M}(\tau_{0,i-1})$ .
    - b. Generate ciphertext and state  $(c_{0,i}, \mathfrak{s}_{0,i}) \leftarrow \Pi_{0,i}^E(\text{msg}, m_{0,i}, \text{pk}_1, \mathfrak{s}_{0,i-1})$ .
    - c. Locally store  $\mathfrak{s}_{0,i}$  and send  $c_{0,i}$  to  $P_1$ .
  2. Then,  $P_1$  performs the following steps:<sup>8</sup>
    - a. Sample “innocuous message”  $m_{1,i} \leftarrow \mathcal{M}(\tau_{1,i-1})$ .
    - b. Generate ciphertext and state  $(c_{1,i}, \mathfrak{s}_{1,i}) \leftarrow \Pi_{1,i}^E(m_{1,i}, \text{pk}_0, \mathfrak{s}_{1,i-1})$ .
    - c. Locally store  $\mathfrak{s}_{1,i}$  and send  $c_{1,i}$  to  $P_0$ .

After  $r$  rounds,  $P_1$  computes  $\text{msg}' = \Pi_{1,\text{out}}^E(\text{sk}_1, \mathfrak{s}_{1,r})$  and halts.
- **Correctness.** For any  $\text{msg} \in \{0, 1\}^\kappa$ , if  $P_0$  and  $P_1$  play  $\Pi^E$  honestly, then  $\text{msg}' = \text{msg}$  with probability  $1 - \text{negl}(\kappa)$ . The probability is taken over the key generation  $(\text{pk}_0, \text{sk}_0), (\text{pk}_1, \text{sk}_1) \leftarrow E.\text{Gen}$  and the randomness of the protocol algorithms, as well as the message distribution  $\mathcal{M}$ .
- **Subliminal Indistinguishability.** For any semantically secure public-key encryption scheme  $E$ , any  $\text{msg} \in \{0, 1\}^\kappa$  and any next-message distribution  $\mathcal{M}$ , for  $(\text{pk}_i, \text{sk}_i) \leftarrow E.\text{Gen}$ ,  $i \in \{0, 1\}$ , the following distributions are computationally indistinguishable:

<u>Ideal</u> ( $\text{pk}_0, \text{sk}_0, \text{pk}_1, \text{sk}_1, \mathcal{M}$ ):	<u>Subliminal<math>_{\Pi}</math></u> ( $\text{msg}, \text{pk}_0, \text{sk}_0, \text{pk}_1, \text{sk}_1, \mathcal{M}$ ):
for $i = 1, \dots, r$ :	for $i = 1, \dots, r$ :
$m_{0,i} \leftarrow \mathcal{M}(\tau_{0,i})$	$m_{0,i} \leftarrow \mathcal{M}(\tau_{0,i})$
$m_{1,i} \leftarrow \mathcal{M}(\tau_{1,i})$	$m_{1,i} \leftarrow \mathcal{M}(\tau_{1,i})$
$c_{0,i} \leftarrow E.\text{Enc}(\text{pk}_1, m_{0,i})$	$(c_{0,i}, \mathfrak{s}_{0,i}) \leftarrow \Pi_{0,i}^E(\text{msg}, m_{0,i}, \text{pk}_1, \mathfrak{s}_{0,i-1})$
$c_{1,i} \leftarrow E.\text{Enc}(\text{pk}_0, m_{1,i})$	$(c_{1,i}, \mathfrak{s}_{1,i}) \leftarrow \Pi_{1,i}^E(m_{1,i}, \text{pk}_0, \mathfrak{s}_{1,i-1})$
output $(\text{pk}_0, \text{sk}_0, \text{pk}_1, \text{sk}_1; (c_{b,i})_{b \in \{0,1\}, i \in [r]})$	output $(\text{pk}_0, \text{sk}_0, \text{pk}_1, \text{sk}_1; (c_{b,i})_{b \in \{0,1\}, i \in [r]})$

If the *subliminal indistinguishability* requirement is satisfied only for next-message distributions  $\mathcal{M}$  in a restricted set  $\mathbb{M}$ , rather than for any  $\mathcal{M}$ , then  $\Pi$  is said to be a *subliminal communication scheme for  $\mathbb{M}$* .

For simplicity, Definition 7 presents a communication scheme in which only a single hidden message  $\text{msg}$  is transmitted. More generally, it is desirable to transmit multiple messages, and bidirectionally, and perhaps in an adaptive manner.<sup>9</sup> In multi-message schemes, it may be beneficial for efficiency that the protocol have a two-phase structure where some initial

<sup>8</sup> Note that the steps executed by  $P_0$  and  $P_1$  are entirely symmetric except in the following two aspects: first,  $P_0$ 's input  $\text{msg}$  is present in step 1b but not in step 2b; and secondly, the state  $\mathfrak{s}_{1,i-1}$  used in step 2b contains the round- $i$  message  $c_{0,i}$ , whereas the state  $\mathfrak{s}_{0,i-1}$  used in step 1b depends only on the transcript until round  $i - 1$ .

<sup>9</sup> That is, the messages to be transmitted may become known as the protocol progresses, rather than all being known at the outset. This is the case, for example, if future messages depend on responses to previous ones.



preprocessing is done in the first phase, and then the second phase can thereafter be invoked many times to transmit different hidden messages.<sup>10</sup> This is a useful notion later in the paper, for our constructions, so we give the definition of a multi-message scheme here.

► **Definition 8.** A *multi-message subliminal communication scheme* is a two-party protocol defined by a pair  $(\Phi, \Xi)$  where  $\Phi$  (“Setup Phase”) and  $\Xi$  (“Communication Phase”) each define a two-party protocol. Each party outputs a state at the end of  $\Phi$ , which it uses as an input in each subsequent invocation of  $\Xi$ . An execution of a multi-message subliminal communication scheme consists of an execution of  $\Phi$  followed by one or more executions of  $\Xi$ . More formally:

$$\begin{aligned}\Phi^E &= (\Phi_{0,1}^E, \Phi_{1,1}^E, \Phi_{0,2}^E, \Phi_{1,2}^E, \dots, \Phi_{0,r}^E, \Phi_{1,r}^E) \\ \Xi^E &= (\Xi_{0,1}^E, \Xi_{1,1}^E, \Xi_{0,2}^E, \Xi_{1,2}^E, \dots, \Xi_{0,r'}^E, \Xi_{1,r'}^E, \Xi_{1,\text{out}}^E)\end{aligned}$$

where  $r, r' \in \text{poly}$  are the number of exchange-rounds in  $\Phi$  and  $\Xi$  respectively. and where each  $\Phi_{b,i}^E, \Xi_{b,i}^E$  is a PPT algorithm with oracle access to the algorithms of a public-key encryption scheme  $E$ . The protocol must satisfy the following syntax, correctness and security guarantees.

■ **Syntax.** In each exchange-round  $i = 1, \dots, r$  of  $\Phi$ :  $P_0$  executes the following steps for  $b = 0$ , and then  $P_1$  executes the same steps for  $b = 1$ .

1. Sample “innocuous message”  $m_{b,i} \leftarrow \mathcal{M}(\tau_{b,i-1})$ .
2. Generate ciphertext and state  $(c_{b,i}, \mathfrak{s}_{b,i}) \leftarrow \Phi_{b,i}^E(m_{b,i}, \text{pk}_{1-b}, \mathfrak{s}_{b,i-1})$ .
3. Locally store  $\mathfrak{s}_{b,i}$  and send  $c_{b,i}$  to  $P_{1-b}$ .

After the completion of  $\Phi$ , either party may initiate  $\Xi$  by sending a first message of the  $\Xi$  protocol (with respect to a message  $\text{msg}$  to be steganographically hidden, known to the initiating party). Let  $P_S$  and  $P_R$  denote the initiating and non-initiating parties in an execution of  $\Xi$ , respectively.<sup>11</sup> Let  $\text{msg} \in \{0, 1\}^\kappa$  be the hidden message that  $P_S$  is to transmit to  $P_R$  in an undetectable fashion during an execution of  $\Xi$ .

The execution of  $\Xi$  proceeds as follows over exchange-rounds  $i' = 1, \dots, r'$ :

- $P_S$  acts as follows:
  1. Sample  $m_{S,r+i'} \leftarrow \mathcal{M}(\tau_{S,r+i'-1})$ .
  2. Generate  $(c_{S,r+i'}, \mathfrak{s}_{S,r+i'}) \leftarrow \Xi_{0,i'}^E(\text{msg}, m_{S,r+i'}, \text{pk}_R, \mathfrak{s}_{S,r+i'-1})$ .
  3. Locally store  $\mathfrak{s}_{S,r+i'}$  and send  $c_{S,r+i'}$  to  $P_R$ .
- $P_R$  acts as follows:
  1. Sample  $m_{R,r+i'} \leftarrow \mathcal{M}(\tau'_{R,r+i'-1})$ .
  2. Generate  $(c_{R,r+i'}, \mathfrak{s}_{R,r+i'}) \leftarrow \Xi_{1,i'}^E(m_{R,r+i'}, \text{pk}_S, \mathfrak{s}_{R,r+i'-1})$ .
  3. Locally store  $\mathfrak{s}_{R,r+i'}$  and send  $c_{R,r+i'}$  to  $P_S$ .

At the end of an execution of  $\Xi$ ,  $P_R$  computes  $\text{msg}' = \Xi_{1,\text{out}}^E(\text{sk}_1, \mathfrak{s}_{1,r+r'})$ .

■ **Correctness.** For any  $\text{msg} \in \{0, 1\}^\kappa$ , if  $P_0$  and  $P_1$  execute  $(\Phi, \Xi)$  honestly, then for every execution of  $\Xi$ , the transmitted and received messages  $\text{msg}$  and  $\text{msg}'$  are equal with overwhelming probability. The probability is taken over the key generation  $(\text{pk}_0, \text{sk}_0), (\text{pk}_1, \text{sk}_1) \leftarrow E.\text{Gen}$  and the randomness of the protocol algorithms, as well as the message distribution  $\mathcal{M}$ .

<sup>10</sup> As a concrete example: consider a protocol for transmitting multiple encrypted messages with a one-time “phase 1” consisting of key exchange, and a “phase 2” encompassing the ciphertext transmission which can be invoked many times.

<sup>11</sup> Subscripts  $S, R \in \{0, 1\}$  stand for “sender” and “receiver,” respectively.

- Subliminal Indistinguishability.** For any semantically secure public-key encryption scheme  $E$ , any polynomial  $p = p(\kappa)$ , any sequence of hidden messages  $\vec{msg} = (msg_i)_{i \in [p]} \in (\{0, 1\}^\kappa)^p$ , any sequence of bits  $\vec{b} = (b_1, \dots, b_p) \in \{0, 1\}^p$  and any next-message distribution  $\mathcal{M}$ , for  $(pk_b, sk_b) \leftarrow E.Gen$ ,  $b \in \{0, 1\}$  the following distributions are computationally indistinguishable:

<p><b>Ideal</b><math>(pk_0, sk_0, pk_1, sk_1, \mathcal{M})</math>:</p> <p>for <math>i = 1, \dots, r + pr'</math>:</p> <p style="padding-left: 20px;"><math>m_{0,i} \leftarrow \mathcal{M}(\tau_{0,i})</math></p> <p style="padding-left: 20px;"><math>m_{1,i} \leftarrow \mathcal{M}(\tau_{1,i})</math></p> <p style="padding-left: 20px;"><math>c_{0,i} \leftarrow E.Enc(pk_1, m_{0,i})</math></p> <p style="padding-left: 20px;"><math>c_{1,i} \leftarrow E.Enc(pk_0, m_{1,i})</math></p> <p>output:</p> <p style="padding-left: 20px;"><math>(pk_0, sk_0, pk_1, sk_1; (c_{b,i})_{b \in \{0,1\}, i \in [r+pr']})</math></p>	<p><b>Subliminal</b><math>_{\Phi, \Xi}(\vec{msg}, \vec{b}, pk_0, sk_0, pk_1, sk_1, \mathcal{M})</math>:</p> <p>for <math>i = 1, \dots, r</math>:</p> <p style="padding-left: 20px;"><math>m_{0,i} \leftarrow \mathcal{M}(\tau_{0,i})</math></p> <p style="padding-left: 20px;"><math>m_{1,i} \leftarrow \mathcal{M}(\tau_{1,i})</math></p> <p style="padding-left: 20px;"><math>(c_{0,i}, s_{0,i}) \leftarrow \Phi_{0,i}^E(msg, m_{0,i}, pk_1, s_{0,i-1})</math></p> <p style="padding-left: 20px;"><math>(c_{1,i}, s_{1,i}) \leftarrow \Phi_{1,i}^E(m_{1,i}, pk_0, s_{1,i-1})</math></p> <p>for <math>j = 1, \dots, p</math>:</p> <p style="padding-left: 20px;">let <math>\beta = b_j</math> and <math>\bar{\beta} = 1 - b_j</math></p> <p style="padding-left: 20px;">for <math>i' = 1, \dots, r'</math>:</p> <p style="padding-left: 40px;">let <math>\iota = r + (j - 1)r' + i'</math></p> <p style="padding-left: 40px;"><math>m_{\beta,\iota} \leftarrow \mathcal{M}(\tau_{\beta,\iota})</math></p> <p style="padding-left: 40px;"><math>m_{\bar{\beta},\iota} \leftarrow \mathcal{M}(\tau_{\bar{\beta},\iota})</math></p> <p style="padding-left: 40px;"><math>(c_{\beta,\iota}, s_{\beta,\iota}) \leftarrow \Xi_{\beta,i'}^E(msg, m_{\beta,\iota}, pk_{\bar{\beta}}, s_{\beta,\iota-1})</math></p> <p style="padding-left: 40px;"><math>(c_{\bar{\beta},\iota}, s_{\bar{\beta},\iota}) \leftarrow \Xi_{\bar{\beta},i'}^E(m_{\bar{\beta},\iota}, pk_{\beta}, s_{\bar{\beta},\iota-1})</math></p> <p>output:</p> <p style="padding-left: 20px;"><math>(pk_0, sk_0, pk_1, sk_1; (c_{b,i})_{b \in \{0,1\}, i \in [r+pr']})</math></p>
---	--

If the *subliminal indistinguishability* requirement is satisfied only for  $M$  in a restricted set  $\mathbb{M}$ , rather than for any  $\mathcal{M}$ , then  $(\Phi, \Xi)$  is said to be a *multi-message subliminal communication scheme* for  $\mathbb{M}$ .

## 4 Impossibility Results

### 4.1 Locally Decodable Subliminal Communication Schemes

A first attempt at achieving subliminal communication might consider schemes with the following natural property: the receiving party  $P_1$  extracts hidden bits *one ciphertext at a time*, by the application of a single decoding function. We refer to such schemes as *locally decodable* and our next impossibility theorem shows that non-trivial locally decodable schemes do not exist if the encryption scheme  $E$  is chosen adversarially.

► **Theorem 9.** *For any locally decodable protocol  $\Pi$  satisfying the syntax of a single-message subliminal communication scheme (Definition 7), there exists a semantically secure public-key encryption scheme  $E$  such that  $E$  violates the correctness condition of Definition 7. Therefore, no locally decodable protocol  $\Pi$  is a subliminal communication scheme.*

► **Remark.** The essence of the above theorem is the impossibility of deterministic extraction: no single deterministic function can deterministically extract from ciphertexts of arbitrary encryption schemes. The way to bypass this impossibility is to have the extractor depend on the encryption scheme. Note that multiple-source extraction, which is used in our constructions in the subsequent sections, implicitly do depend on the underlying encryption scheme, since the additional sources of input depend on the encryption scheme and thus can be thought of as “auxiliary input” that is specific to the encryption scheme at hand.

## 4.2 Steganography for Adversarial Cover Distributions

Our second impossibility result concerns a much more general class of communication schemes, which we call *steganographic communication schemes*. Subliminal communication schemes, as well as the existing notions of public-key steganography and steganographic key exchange from the steganography literature, are instantiations of the more general definition of a (multi-message) steganographic communication scheme. To our knowledge, the general notion of a steganographic communication scheme has not been formalized in this way in prior work. In the context of this work, the general definition is helpful for proving broad impossibilities across multiple types of steganographic schemes.

As mentioned in the introduction, a limitation of all existing results in the steganographic literature, to our knowledge, is that they assume that the *cover distribution* – i.e., the distribution of innocuous objects in which steganographic communication is to be embedded – is fixed *a priori*. In particular, the cover distribution is assumed not to depend on the description of the steganographic communication scheme. The impossibility result given in Section 4.1 is an example illustrative of the power of adversarially choosing the cover distribution: Theorem 9 says that by choosing the encryption scheme  $E$  to depend on a given subliminal communication scheme, an adversary can rule out the possibility of any hidden communication at all.

Our next impossibility result (Theorem 10) shows that if the cover distribution is chosen adversarially, then non-trivial steganographic communication is impossible.

► **Theorem 10.** *Let  $\Pi$  be a protocol with the syntax of a steganographic communication scheme. Then for any  $k \in \mathbb{N}$ , there exists a cover distribution  $\mathcal{C}$  of conditional min-entropy  $k$  such that steganographic indistinguishability of  $\Pi$  does not hold.*

We have elected to present the definition of a *steganographic communication scheme* as well as the proof of Theorem 10 in the full version of this paper [17] since the definition introduces a set of new notation only used for the corresponding impossibility result, and both the definition and the impossibility result are somewhat tangential to the main results of this work, whose focus is on subliminal communication schemes.

## 5 Construction of the Subliminal Scheme

The goal of this section is to establish the following theorem, which states that our construction  $(\Phi^*, \Xi^*)$  is a subliminal communication scheme when instantiated with a pseudorandom key-exchange protocol (such as Diffie-Hellman).

► **Theorem 11.** *The protocol  $(\Phi^*, \Xi^*)$  given in Definition 17, when instantiated with a pseudorandom key-exchange protocol  $\Lambda$ , is a multi-message subliminal communication scheme.*

The description of our scheme can be found in the following subsections. Our construction makes no assumption on the message distribution  $\mathcal{M}$  and in particular holds when the exchanged plaintexts (of the adversarial encryption scheme  $E$ ) are a fixed, adversarially chosen sequence of messages. An informal outline of the construction is given next.

► **Definition 12.** Outline of the construction.

### 1. Setup Phase $\Phi^*$

- a. A  $\tilde{O}(\log \kappa)$ -bit string  $S$  is established between  $P_0$  and  $P_1$  by extracting randomness from pairs of consecutive ciphertexts. (*Protocol overview in Section 5.1.*)

- b. Let  $\text{Ext}$  be a strong seeded extractor, and let  $S$  serve as its seed. By rejection-sampling ciphertexts  $c$  until  $\text{Ext}_S(c) = \text{str}$ , either party can embed a random string  $\text{str}$  of their choice in the conversation. (*Protocol overview in Section 5.2.*) By embedding in this manner the messages of a pseudorandom key-exchange protocol, both parties establish a shared secret  $\text{sk}^*$ .<sup>12</sup>

## 2. Communication Phase $\Xi^*$

Both parties can now communicate arbitrary messages of their choice by (1) encrypting them using a pseudorandom secret-key encryption scheme  $\text{SKE}$  using  $\text{sk}^*$  as the secret key, and (2) embedding the ciphertexts of  $\text{SKE}$  using the rejection-sampling technique described in Step 1b.<sup>13</sup> (*Detailed protocol in Section 5.3.*)

The full protocol is given in Section 5.3.

## 5.1 Establishing a Shared Seed

In this section, we give a protocol which allows  $P_0$  and  $P_1$  to establish a random public parameter which will be used in subsequent phases of our subliminal scheme. This can be thought of as drawing a subliminal scheme at random from a family of subliminal schemes. The parameter is public in the sense that any eavesdropper gains knowledge of it. A crucial point is that the random draw occurs *after* the adversarial encryption scheme  $\text{E}$  is fixed, thus bypassing the impossibility results of Section 4.

Our strategy is simple: extract randomness from pairs of ciphertexts. Since the extractor does not receive the key, semantic security holds with respect to the extractor: a pair of ciphertexts for two arbitrary messages is indistinguishable from two encryptions of a fixed message; thus, a same-source extractor suffices for our purposes. Even though semantic security guarantees only  $\omega(\log \kappa)$  min-entropy of ciphertexts, we can use the “greater-than” extractor (Definition 13) applied to pairs of ciphertexts, and obtain Theorem 14.

► **Definition 13.** The *greater-than extractor*  $\text{GT}$  is defined by  $\text{GT}(x, y) = \mathbf{1}[x \geq y]$ .

► **Theorem 14.** Let  $\text{E}$  be a semantically secure public-key encryption scheme and let  $\rho \in \text{poly}$ . Define random variables as follows.

- For  $b \in \{0, 1\}$ , let  $K_b = (PK_b, SK_b) = \text{E.Gen}(1^\kappa)$ .
- For  $b \in \{0, 1\}$  and  $i \in [2\rho]$ , let  $C_{b,i} = \text{E.Enc}(PK_{1-b}, m_{b,i})$  representing the ciphertexts exchanged between  $P_0$  and  $P_1$  during  $2\rho$  exchange-rounds.
- Let  $S = (\text{GT}(C_{0,1}, C_{0,2}), \text{GT}(C_{1,1}, C_{1,2}), \dots, \text{GT}(C_{1,2\rho-1}, C_{1,2\rho}))$ .

There exists a negligible function  $\varepsilon$  such that  $\|(K_0, K_1, S) - (K_0, K_1, U_{2\rho})\|_s \leq \varepsilon$ .

## 5.2 Embedding Random Strings

In this section, we assume that both parties have access to a public parameter  $S$  and construct a protocol which allows for embedding of uniformly random strings into ciphertexts of an adversarially chosen encryption scheme  $\text{E}$ , as required by Steps 1b and 2 of the construction outline (Definition 12). The security guarantee is that for a uniformly random parameter  $S$  and uniformly random strings to be embedded, the ciphertexts of  $\text{E}$  with embedded random

<sup>12</sup> An eavesdropper who knows  $S$  can compute  $\text{str}$ . Nonetheless, (1)  $\text{sk}^*$  is unknown to the eavesdropper by the security of the key-exchange protocol and (2) the transcript is indistinguishable from one in which no key exchange occurred at all, due to the pseudorandomness of the key-exchange messages.

<sup>13</sup> Again, an eavesdropper could know the  $\text{SKE}$  ciphertexts exchanged, if he knew the seed  $S$ , but could not distinguish the  $\text{SKE}$  ciphertexts from truly random strings, and thus could not tell whether any subliminal communication was occurring at all. See also footnote 12.

**Algorithm 1** Rejection sampler  $\Sigma^{E,S}$ .PUBLIC PARAMETER:  $S$  (a  $d$ -bit seed).INPUT:  $(\text{str}, m, \text{pk})$  where  $\text{str}$  is the string to be embedded.

1. Generate encryption  $c \leftarrow \text{E.Enc}(\text{pk}, m)$ .
2. If  $\text{Ext}(S, c) = \text{str}$ , then output  $c$ . Else, go back to step 1.

strings are indistinguishable from ciphertexts of  $\text{E}$  produced by direct application of  $\text{E.Enc}$ , even to an adversary who knows the decryption keys of  $\text{E}$ . This can be thought of as a relaxation of subliminal indistinguishability (Definition 7) where the two main differences are that (1) the parties have shared knowledge of a random seed, and (2) indistinguishability only holds when embedding a *random* string, rather than for arbitrary strings. Our construction (Theorem 15) relies on a strong seeded extractor that can extract logarithmically many bits from sources of super-logarithmic min-entropy, we note that almost universal hashing is a simple such extractor.

► **Theorem 15.** *Let  $\text{Ext} : \{0, 1\}^d \times \{0, 1\}^n \rightarrow \{0, 1\}^v$  be a strong seeded extractor for super-logarithmic min-entropy with  $v = O(\log \kappa)$ , and let  $\text{E}$  be a semantically secure encryption scheme with ciphertext space  $\mathcal{C} = \{0, 1\}^n$ . Let  $\Sigma^{E,S}$  be defined as in Algorithm 1. Then the following guarantees hold:*

1. Correctness: for any  $S \in \{0, 1\}^d$  and  $\text{str} \in \{0, 1\}^v$ , if  $c = \Sigma^{E,S}(\text{str}, m, \text{pk})$ , and  $\text{str}' = \text{Ext}(S, c)$ , then  $\text{str}' = \text{str}$ .
2. Security: let  $(PK, SK) = \text{E.Gen}(1^\kappa)$ ,  $C = \text{E.Enc}(PK, m)$  and  $C' = \Sigma^{E,U_d}(U_v, m, PK)$ ; then  $\|(PK, SK, U_d, C) - (PK, SK, U_d, C')\|_s \leq \varepsilon(\kappa)$  for some negligible function  $\varepsilon$ .

► **Remark.** Rejection sampling is a simple and natural approach that has been used by prior work in the steganographic literature, such as [2]. Despite the shared use of this common technique, our construction is more different from prior art than it might seem at first glance. The novelty of our construction arises from the challenges of working in a model with a stronger adversary who can choose the distribution of ciphertexts (i.e., the adversary gets to choose the public-key encryption scheme  $\text{E}$ ). We manage to bypass the impossibilities outlined in Section 4 notwithstanding this stronger adversarial model, and in contrast to prior work, construct a protocol to established a shared seed from scratch, rather than simply assuming that one has been established in advance.

### 5.3 Full Protocol $(\Phi^*, \Xi^*)$

► **Definition 16** (Key-exchange protocol syntax). A key-exchange protocol is a two-party protocol defined by  $\Lambda = (\Lambda_{0,1}, \Lambda_{1,1}, \Lambda_{0,2}, \Lambda_{1,2}, \dots, \Lambda_{0,k}, \Lambda_{1,k}, \Lambda_{0,\text{out}}, \Lambda_{1,\text{out}})$ . We assume  $k$  simultaneous communication rounds, where  $\Lambda_{b,i}$  represents the computation performed by  $P_b$  in the  $i$ th round. The parties are stateful and their state is implicitly updated at each round to contain the transcript so far and any local randomness generated so far. Each  $\Lambda_{b,i}$  takes as input the transcript up to round  $i - 1$  and the state of  $P_b$ , and outputs a message  $\lambda_{b,i}$  to be sent in the  $i$ th round. For notational simplicity, we write explicitly only the first input to  $\Lambda_{b,i}$ , and leave the second input (i.e., the state) implicit.  $\Lambda_{0,\text{out}}, \Lambda_{1,\text{out}}$  are run by  $P_0, P_1$  respectively to compute the shared secret at the conclusion of the protocol.

Next, we give the full construction of  $(\Phi^*, \Xi^*)$  following the outline in Definition 12.

► **Definition 17.**  $(\Phi^*, \Xi^*)$  is parametrized by the following.

- $\text{Ext} : \{0, 1\}^d \times \{0, 1\}^n \rightarrow \{0, 1\}^v$ , a strong seeded extractor.
  - $\Lambda$ , a pseudorandom key-exchange protocol with  $\ell$ -bit messages.<sup>14</sup>
  - SKE, a pseudorandom secret key encryption scheme with  $\xi$ -bit ciphertexts.
- We define each phase of our construction in turn.

## 1. Setup Phase $\Phi^*$

### a. Establishing a $d$ -bit shared seed

- For  $b \in \{0, 1\}$  and  $i \in \{1, \dots, d\}$ ,  $\Phi_{b,i}^*(m_{b,i}, \text{pk}_{1-b}, \mathfrak{s}_{b,i-1})$  outputs a ciphertext  $c_{b,i} = \text{E.Enc}(\text{pk}_{1-b}, m_{b,i})$  and sets the updated state  $\mathfrak{s}_{b,i}$  to be the transcript of all protocol messages sent and received so far.
- At the conclusion of the  $d$  exchange-rounds, each party updates his state to contain the seed  $S$  which is defined by

$$S = (\text{GT}(c_{0,1}, c_{0,2}), \text{GT}(c_{1,1}, c_{1,2}), \dots, \text{GT}(c_{1,d-1}, c_{1,d})) .$$

This seed  $S$  is assumed to be accessible in all future states throughout both phases during the remainder of the protocol.

### b. Subliminal key exchange

Let  $\nu = \frac{\ell}{v}$ . Subliminal key exchange occurs over  $k \cdot \nu$  exchange-rounds.

- For  $j \in \{1, \dots, k\}$  and  $b \in \{0, 1\}$ :
  - $P_b$  retrieves from his state the key-exchange transcript so far  $(\lambda_{b,j'})_{b \in \{0,1\}, j' < j}$ .
  - $P_b$  computes the next key-exchange message  $\lambda_{b,j} \leftarrow \Lambda_{b,j}((\lambda_{b,j'})_{b \in \{0,1\}, j' < j})$ .
  - $P_b$  breaks  $\lambda_{b,j}$  into  $v$ -bit blocks  $\lambda_{b,j} = \lambda_{b,j}^1 || \dots || \lambda_{b,j}^\nu$ .
  - The  $\nu$  blocks are transmitted sequentially as follows. For  $\iota \in \{1, \dots, \nu\}$ :
    - Let  $i = d + (j - 1)\nu + \iota$ .
    - $\Phi_{b,i}^*(m_{b,i}, \text{pk}_{1-b}, \mathfrak{s}_{b,i-1})$  outputs  $c_{b,i} \leftarrow \Sigma^{\text{E},S}(\lambda_{b,j}^\iota, m_{b,i}, \text{pk}_{1-b})$  and sets the updated state  $\mathfrak{s}_{b,i}$  to contain the transcript of all protocol messages sent and received so far.
- At the conclusion of the  $\iota$  exchange-rounds, each party  $b \in \{0, 1\}$  updates his state to contain the  $j$ th key-exchange message  $\lambda_{1-b,j}$  computed as follows:

$$\lambda_{1-b,j} = \text{Ext}(S, c_{b,d+(j-1)\nu+1}) || \dots || \text{Ext}(S, c_{b,d+j\nu})$$

- At the conclusion of the  $k \cdot \nu$  exchange rounds, each party updates his state to contain the secret key  $\text{sk}^*$  computed as:  $\text{sk}^* = \text{SKE.Gen}(1^\kappa; \Lambda_{\text{out}}((\lambda_{b,j})_{b \in \{0,1\}, j \in [k]}))$ .

## 2. Communication Phase $\Xi^*$

Each communication phase occurs over  $r' = \xi/v$  exchange-rounds.

Let  $\beta \in \{0, 1\}$  be the initiating party and let  $\bar{\beta} = 1 - \beta$ .

$P_\beta$  performs the following steps.

- Generate  $c^* \leftarrow \text{SKE.Enc}(\text{sk}^*, \text{msg})$ .
  - Break  $c^*$  into  $v$ -bit blocks  $c^* = c_1^* || \dots || c_{r'}^*$ .
- For  $i' \in \{1, \dots, r'\}$ :
- Let  $i'' = r + i'$ .
  - $\Xi_{0,i'}^*(\text{msg}, m_{0,i''}, \text{pk}_{\bar{\beta}}, \mathfrak{s}_{\beta,i''-1})$  outputs  $c_{\beta,i''} \leftarrow \Sigma^{\text{E},S}(c_{i'}^*, m_{\beta,i''}, \text{pk}_{\bar{\beta}})$ .
  - $\Xi_{1,i'}^*(m_{\bar{\beta},i''}, \text{pk}_{\beta}, \mathfrak{s}_{\bar{\beta},i''-1})$  outputs  $c_{\bar{\beta},i''} \leftarrow \text{E.Enc}(\text{pk}_{\beta}, m_{\bar{\beta},i''})$ .

<sup>14</sup>In presenting our construction  $(\Phi^*, \Xi^*)$ , we do not denote the state of parties w.r.t. the key-exchange protocol  $\Lambda$  by a separate variable, but assume that it is part of the state  $\mathfrak{s}_{b,i}$  of the overall protocol.

- 37     ■ Both parties update their state to contain the transcript of all protocol messages  
38         exchanged so far.

39     After the  $r'$  exchange-rounds,  $P_{\tilde{\beta}}$  computes  $c^{**} = \text{Ext}(S, c_{\beta, r+1}) \parallel \dots \parallel \text{Ext}(S, c_{\beta, r+r'})$ .  
40     Then,  $P_{\tilde{\beta}}$  outputs  $\text{msg}' \leftarrow \text{SKE.Dec}(\text{sk}^*, c^{**})$ . (That is,  $\Xi_{1, \text{out}}^*(\mathfrak{s}_{\tilde{\beta}, r'}) = \text{msg}'$ .)

## 6 Open problems

**Deterministic Extraction.** Our impossibility result in Theorem 9 holds because the adversary can choose the encryption scheme  $\mathbf{E}$  as a function of a given candidate subliminal scheme. However, note that under the additional assumption that  $\mathbf{E}$  is restricted to a predefined class  $E$  of encryption schemes, we could bypass this impossibility as long as a deterministic extractor that can extract randomness from ciphertexts of any encryption scheme in  $E$  exists. We are only aware of two deterministic extractors leading to a positive result for restricted classes of encryption schemes:

- if an upper bound on the circuit size of  $\mathbf{E}$  is known, then we can use the deterministic extractor from [21]. This extractor relies on strong complexity-theoretic assumptions and requires the sources to have min-entropy  $(1 - \gamma)n$  for some unspecified constant  $\gamma$ .
- if  $\mathbf{E}$  is computed by a circuit of constant depth ( $\mathbf{AC}^0$ ), then the deterministic extractor of [22] can be used and requires  $\sqrt{n}$  min-entropy.

Both these extractors have a min-entropy requirement which is not satisfied by ciphertexts of arbitrary encryption schemes. However, it would be interesting to consider improved constructions for the case of specific encryption schemes, or to consider extractors specifically for encryption circuits as opposed to arbitrary circuits satisfying a min-entropy requirement. This would also have direct implications for the efficiency of the subliminal scheme of Section 5.2: indeed, one could then skip Step 1a and use a deterministic extractor directly in Steps 1b and 2, thus saving  $\tilde{O}(\log \kappa)$  exchange-rounds in the setup phase.

**Multi-Source Extraction.** Another interesting question is whether multi-source extractors for the specific case when the sources are independent and identically distributed can achieve better parameters than extractors for general independent sources. We already saw that a very simple extractor (namely, the “greater-than” function) works for i.i.d. sources and extracts one bit with negligible bias, even when the sources only have  $\omega(\log \kappa)$  min-entropy. The non-constructive result of [9] guarantees the existence of a two-source extractor of negligible bias and output length  $\omega(\log \kappa)$  for sources of min-entropy  $\omega(\log \kappa)$ . However, known *explicit* constructions are far from achieving the same parameters, and improving them in the specific case of identically distributed sources is an interesting open problem which was also mentioned in [3].

---

## References

- 1 Per Austrin, Kai-Min Chung, Mohammad Mahmoody, Rafael Pass, and Karn Seth. On the Impossibility of Cryptography with Tamperable Randomness. In *CRYPTO 2014, Proceedings, Part I*, volume 8616 of *Lecture Notes in Computer Science*, pages 462–479. Springer, 2014. doi:10.1007/978-3-662-44371-2\_26.
- 2 Michael Backes and Christian Cachin. Public-Key Steganography with Active Attacks. In *TCC 2005, Proceedings*, volume 3378 of *Lecture Notes in Computer Science*, pages 210–226. Springer, 2005. doi:10.1007/978-3-540-30576-7\_12.



- 3 Boaz Barak, Russell Impagliazzo, and Avi Wigderson. Extracting Randomness Using Few Independent Sources. In *45th Symposium on Foundations of Computer Science (FOCS 2004), Proceedings*, pages 384–393, 2004. doi:10.1109/FOCS.2004.29.
- 4 Mihir Bellare, Joseph Jaeger, and Daniel Kane. Mass-surveillance without the State: Strongly Undetectable Algorithm-Substitution Attacks. In Indrajit Ray, Ninghui Li, and Christopher Kruegel, editors, *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security (CCS) 2015*, pages 1431–1440. ACM, 2015. doi:10.1145/2810103.2813681.
- 5 Mihir Bellare, Kenneth G. Paterson, and Phillip Rogaway. Security of Symmetric Encryption against Mass Surveillance. In *CRYPTO 2014, Proceedings, Part I*, volume 8616 of *Lecture Notes in Computer Science*, pages 1–19. Springer, 2014. doi:10.1007/978-3-662-44371-2\_1.
- 6 Sebastian Berndt and Maciej Liśkiewicz. Algorithm Substitution Attacks from a Steganographic Perspective. In Bhavani M. Thuraisingham, David Evans, Tal Malkin, and Dongyan Xu, editors, *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security, CCS 2017, Dallas, TX, USA, October 30 - November 03, 2017*, pages 1649–1660. ACM, 2017. doi:10.1145/3133956.3133981.
- 7 Christian Cachin. An Information-Theoretic Model for Steganography. In David Aucsmith, editor, *Information Hiding, Second International Workshop, Portland, Oregon, USA, April 14-17, 1998, Proceedings*, volume 1525 of *Lecture Notes in Computer Science*, pages 306–318. Springer, 1998. doi:10.1007/3-540-49380-8\_21.
- 8 Eshan Chattopadhyay and David Zuckerman. Explicit two-source extractors and resilient functions. In *Proceedings of the 48th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2016*, pages 670–683, 2016.
- 9 Benny Chor and Oded Goldreich. Unbiased Bits from Sources of Weak Randomness and Probabilistic Communication Complexity. *SIAM J. Comput.*, 17(2):230–261, 1988. doi:10.1137/0217015.
- 10 Aloni Cohen and Saleet Klein. The GGM Function Family Is a Weakly One-Way Family of Functions. In *Theory of Cryptography - 14th International Conference, TCC 2016-B, Part I*, volume 9985 of *Lecture Notes in Computer Science*, pages 84–107, 2016. doi:10.1007/978-3-662-53641-4\_4.
- 11 Whitfield Diffie and Martin E. Hellman. New directions in cryptography. *IEEE Trans. Information Theory*, 22(6):644–654, 1976. doi:10.1109/TIT.1976.1055638.
- 12 Yevgeniy Dodis, Ariel Elbaz, Roberto Oliveira, and Ran Raz. Improved Randomness Extraction from Two Independent Sources. In *7th International Workshop on Approximation Algorithms for Combinatorial Optimization Problems, APPROX 2004, and 8th International Workshop on Randomization and Computation, RANDOM 2004, Proceedings*, pages 334–344, 2004.
- 13 Yevgeniy Dodis, Chaya Ganesh, Alexander Golovnev, Ari Juels, and Thomas Ristenpart. A Formal Treatment of Backdoored Pseudorandom Generators. In Elisabeth Oswald and Marc Fischlin, editors, *EUROCRYPT 2015, Proceedings, Part I*, volume 9056 of *Lecture Notes in Computer Science*, pages 101–126. Springer, 2015. doi:10.1007/978-3-662-46800-5\_5.
- 14 Oded Goldreich. The ggm construction does not yield correlation intractable function ensembles. In Oded Goldreich, editor, *Studies in Complexity and Cryptography*, pages 98–108. Springer-Verlag, Berlin, Heidelberg, 2011. URL: <http://dl.acm.org/citation.cfm?id=2028116.2028129>.
- 15 Oded Goldreich, Shafi Goldwasser, and Silvio Micali. How to construct random functions. *J. ACM*, 33(4):792–807, 1986. doi:10.1145/6490.6503.

- 16 Nicholas J. Hopper, John Langford, and Luis von Ahn. Provably Secure Steganography. In *CRYPTO 2002, Santa Barbara, California, USA, Proceedings*, volume 2442 of *Lecture Notes in Computer Science*, pages 77–92. Springer, 2002. doi:10.1007/3-540-45708-9\_6.
- 17 Thibaut Horel, Sunoo Park, Silas Richelson, and Vinod Vaikuntanathan. How to Subvert Backdoored Encryption: Security Against Adversaries that Decrypt All Ciphertexts. Cryptology ePrint Archive, Report 2018/212, 2018. URL: <https://eprint.iacr.org/2018/212>.
- 18 Silvio Micali. ALGORAND: the efficient and democratic ledger. *CoRR*, abs/1607.01341, 2016. arXiv:1607.01341.
- 19 Thomas Mittelholzer. An Information-Theoretic Approach to Steganography and Watermarking. In *Information Hiding, Third International Workshop, IH'99, Proceedings*, volume 1768 of *Lecture Notes in Computer Science*, pages 1–16. Springer, 1999. doi:10.1007/10719724\_1.
- 20 Gustavus J. Simmons. The Prisoners' Problem and the Subliminal Channel. In *CRYPTO 1983*, pages 51–67. Plenum Press, New York, 1983.
- 21 Luca Trevisan and Salil P. Vadhan. Extracting Randomness from Samplable Distributions. In *FOCS 2000, Redondo Beach, California, USA*, pages 32–42, 2000.
- 22 Emanuele Viola. Extractors for Circuit Sources. In *IEEE 52nd Annual Symposium on Foundations of Computer Science, FOCS 2011*, pages 220–229. IEEE Computer Society, 2011. doi:10.1109/FOCS.2011.20.
- 23 Luis von Ahn and Nicholas J. Hopper. Public-Key Steganography. In *EUROCRYPT 2004, Proceedings*, volume 3027 of *Lecture Notes in Computer Science*, pages 323–341. Springer, 2004. doi:10.1007/978-3-540-24676-3\_20.
- 24 Adam L. Young and Moti Yung. Cryptovirology: Extortion-Based Security Threats and Countermeasures. In *1996 IEEE Symposium on Security and Privacy, May 6-8, 1996, Oakland, CA, USA*, pages 129–140. IEEE Computer Society, 1996. doi:10.1109/SECPRI.1996.502676.
- 25 Adam L. Young and Moti Yung. The Dark Side of “Black-Box” Cryptography, or: Should We Trust Capstone? In *CRYPTO 1996, Proceedings*, volume 1109 of *Lecture Notes in Computer Science*, pages 89–103. Springer, 1996. doi:10.1007/3-540-68697-5\_8.
- 26 Adam L. Young and Moti Yung. Kleptography: Using Cryptography Against Cryptography. In *EUROCRYPT '97, Proceedings*, volume 1233 of *Lecture Notes in Computer Science*, pages 62–74. Springer, 1997. doi:10.1007/3-540-69053-0\_6.
- 27 Jan Zöllner, Hannes Federrath, Herbert Klimant, Andreas Pfitzmann, Rudi Piotraschke, Andreas Westfeld, Guntram Wicke, and Gritta Wolf. Modeling the Security of Steganographic Systems. In David Aucsmith, editor, *Information Hiding, Second International Workshop, Portland, Oregon, USA, April 14-17, 1998, Proceedings*, volume 1525 of *Lecture Notes in Computer Science*, pages 344–354. Springer, 1998. doi:10.1007/3-540-49380-8\_24.

# On Integer Programming and Convolution

**Klaus Jansen**

Department of Computer Science, Kiel University, Kiel, Germany  
kj@informatik.uni-kiel.de

**Lars Rohwedder**

Department of Computer Science, Kiel University, Kiel, Germany  
lro@informatik.uni-kiel.de

---

## Abstract

Integer programs with a constant number of constraints are solvable in pseudo-polynomial time. We give a new algorithm with a better pseudo-polynomial running time than previous results. Moreover, we establish a strong connection to the problem  $(\min, +)$ -convolution.  $(\min, +)$ -convolution has a trivial quadratic time algorithm and it has been conjectured that this cannot be improved significantly. We show that further improvements to our pseudo-polynomial algorithm for any fixed number of constraints are equivalent to improvements for  $(\min, +)$ -convolution. This is a strong evidence that our algorithm's running time is the best possible. We also present a faster specialized algorithm for testing feasibility of an integer program with few constraints and for this we also give a tight lower bound, which is based on the SETH.

**2012 ACM Subject Classification** Theory of computation  $\rightarrow$  Integer programming, Theory of computation  $\rightarrow$  Dynamic programming

**Keywords and phrases** Integer programming, convolution, dynamic programming, SETH

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2019.43

**Funding** Research was supported by German Research Foundation (DFG) projects JA 612/20-1 and JA 612/16-1.

## 1 Introduction

Vectors  $v^{(1)}, \dots, v^{(n)} \in \mathbb{R}^m$  that sum up to 0 can be seen as a circle in  $\mathbb{R}^m$  that walks from 0 to  $v^{(1)}$  to  $v^{(1)} + v^{(2)}$ , etc. until it reaches  $v^{(1)} + \dots + v^{(n)} = 0$  again. The Steinitz Lemma [17] says that if each of the vectors is small with respect to some norm, we can reorder them in a way that each point in the circle is not far away from 0 w.r.t. the same norm.

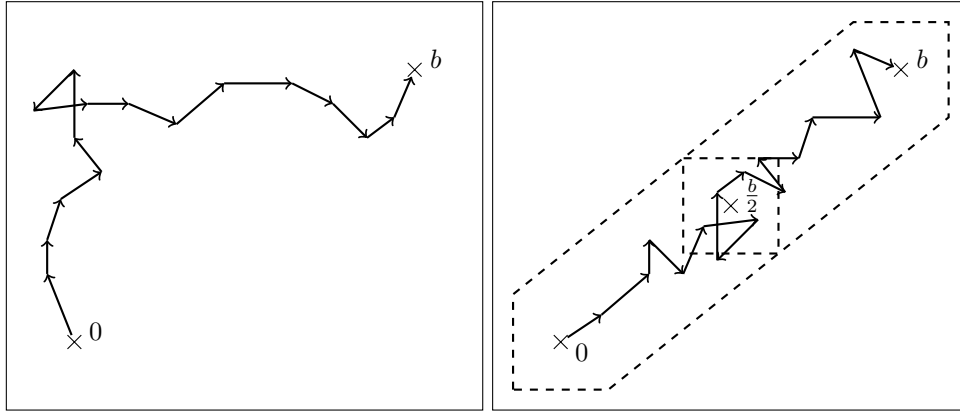
Recently Eisenbrand and Weismantel found a beautiful application of this lemma in the area of integer programming [8]. They looked at ILPs of the form  $\max\{c^T x : Ax = b, x \in \mathbb{Z}_{\geq 0}^n\}$ , where  $A \in \mathbb{Z}^{m \times n}$ ,  $b \in \mathbb{Z}^m$  and  $c \in \mathbb{Z}^n$  and obtained a pseudo-polynomial algorithm in  $\Delta$ , the biggest absolute value of an entry in  $A$ , when  $m$  is treated as a constant. The running time they achieve is  $n \cdot O(m\Delta)^{2m} \cdot \|b\|_1^2$  for finding the optimal solution and  $n \cdot O(m\Delta)^m \cdot \|b\|_1$  for finding only a feasible solution. This improves on a classic algorithm by Papadimitriou, which has a running time of  $O(n^{2m+2} \cdot (m\Delta + m\|b\|_\infty)^{(m+1)(2m+1)})$  [15]. The basic idea in [8] is that a solution  $x^*$  for the ILP above can be viewed as a walk in  $\mathbb{Z}^m$  starting at 0 and ending at  $b$ . Every step is a column of the matrix  $A$ : For every  $i \in \{1, \dots, n\}$  we step  $x_i^*$  times in the direction of  $A_i$  (see left picture in Figure 1). By applying the Steinitz Lemma they show that there is an ordering of these steps such that the walk never strays off far from the direct line between 0 and  $b$  (see right picture in Figure 1). They construct a directed graph with one vertex for every integer point near the line between 0 and  $b$  and create an



© Klaus Jansen and Lars Rohwedder;  
licensed under Creative Commons License CC-BY  
10th Innovations in Theoretical Computer Science (ITCS 2019).  
Editor: Avrim Blum; Article No. 43; pp. 43:1–43:17



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



■ **Figure 1** Steinitz Lemma in Integer Programming.

edge from  $u$  to  $v$ , if  $v - u$  is a column in  $A$ . The weight of the edge is the same as the  $c$ -value of the column. An optimal solution to the ILP can now be obtained by finding a longest path from  $0$  to  $b$ . This can be done in the mentioned time, if one is careful with circles.

In this paper, we present an alternative way to apply the Steinitz Lemma to the same problem. Our approach does not reduce to a longest path problem, but rather solves the ILP in a divide and conquer fashion. Using the Steinitz Lemma and the intuition of a walk from  $0$  to  $b$ , we notice that this walk has to visit a vector  $b'$  near  $b/2$  at some point. We guess this vector and solve the problem with  $Ax = b'$  and  $Ax = b - b'$  independently. Both results can be merged to a solution for  $Ax = b$ . In the sub-problems the norm of  $b$  and the norm of the solution are roughly divided in half. We use this idea in a dynamic program and speed up the process of merging solutions using algorithms for convolution problems. This approach gives us better running times for both the problem of finding optimal solutions and for testing feasibility only. We complete our study by giving (almost) tight conditional lower bounds on the running time in which such ILPs can be solved.

## 1.1 Detailed description of results

In the running times we give, we frequently use logarithmic factors like  $\log(k)$  for some parameter  $k$ . To handle the values  $k \in \{0, 1\}$  formally correct, we would need to write  $\log(k + 1) + 1$  instead of  $\log(k)$  everywhere. This is ignored for simplicity of notation. We are assuming the word RAM model with word size  $O(m \log(m\Delta) + \log(\|b\|_\infty) + \log(\|c\|_\infty))$  (see Preliminaries for details).

### Optimal solutions for ILPs

We show that a solution to  $\max\{c^T x : Ax = b, x \in \mathbb{Z}_{\geq 0}^n\}$  can be found in time  $O(m\Delta)^{2m} \cdot \log(\|b\|_\infty) + O(nm)$ . If given a vertex solution to the fractional relaxation, we can even get to  $O(m\Delta)^{2m} + O(nm)$ . The running time can be improved if there exists a truly sub-quadratic algorithm for (min, +)-convolution (see Section 4.1 for details on the problem). However, it has been conjectured that no such algorithm exists and this conjecture is the base of several lower bounds in fine-grained complexity [7, 14, 3]. We show that for every  $m$  the running time above is essentially the best possible unless the (min, +)-convolution conjecture is false. More formally, for every  $m$  there exists no algorithm that solves ILP in time  $f(m) \cdot (n^{2-\delta} + (\Delta + \|b\|_\infty)^{2m-\delta})$  for some  $\delta > 0$  and an arbitrary computable function  $f$ ,

unless there exists a truly sub-quadratic algorithm for  $(\min, +)$ -convolution. Indeed, this means there is an equivalence between improving algorithms for  $(\min, +)$ -convolution and for ILPs of fixed number of constraints. It is notable that this also rules out improvements when both  $\Delta$  and  $\|b\|_\infty$  are small. Our lower bound does leave open some trade-off between  $n$  and  $O(m\Delta)^m$  like for example  $n \cdot O(m\Delta)^m \cdot \log(\|b\|_\infty)$ , which would be an interesting improvement for sparse instances, i.e., when  $n \ll (2\Delta + 1)^m$ . A running time of  $n^{f(m)} \cdot (m\Delta + m\|b\|_\infty)^{m-\delta}$ , however, is not possible (see feasibility below).

### Feasibility of ILPs

Testing only the feasibility of an ILP is easier than finding an optimal solution. It can be done in time  $O(m\Delta)^m \cdot \log(\Delta) \cdot \log(\Delta + \|b\|_\infty) + O(nm)$  by solving a Boolean convolution problem that has a more efficient algorithm than the  $(\min, +)$ -convolution problem that arises in the optimization version. Under the STRONG EXPONENTIAL TIME HYPOTHESIS (SETH) this running time is tight except for logarithmic factors. If this conjecture holds, there is no  $n^{f(m)} \cdot (m\Delta + m\|b\|_\infty)^{m-\delta}$  time algorithm for any  $\delta > 0$  and any computable function  $f$ .

## 1.2 Other related work

The case where the number of variables  $n$  is fixed and not  $m$  as in this paper behaves somewhat differently. There is a  $2^{O(n \log(n))} \cdot |I|^{O(1)}$  time algorithm ( $|I|$  being the encoding length of the input), whereas an algorithm of the kind  $f(m) \cdot |I|^{O(1)}$  (or even  $|I|^{f(m)}$ ) is impossible for any computable function  $f$ , unless  $P = NP$ . This can be seen with a trivial reduction from UNBOUNDED KNAPSACK (where  $m = 1$ ). The  $2^{O(n \log(n))} \cdot |I|^{O(1)}$  time algorithm is due to Kannan [12] improving over a  $2^{O(n^2)} \cdot |I|^{O(1)}$  time algorithm by Lenstra [11]. It is a long open question whether  $2^{O(n)} \cdot |I|^{O(1)}$  is possible instead [8].

Another intriguing question is whether a running time like  $(m\Delta + m\|b\|_\infty)^{O(m)} \cdot n^{O(1)}$  is still possible when upper bounds on variables are added to the ILP. In [8] an algorithm for this extension is given, but the exponent of  $\Delta$  is  $O(m^2)$ .

As for other lower bounds on pseudo-polynomial algorithms for integer programming, the only result we are aware of is a bound of  $n^{o(m/\log(m))} \cdot \|b\|_\infty^{o(m)}$  due to Fomin et al. [9], which is based on the ETH (a weaker conjecture than the SETH). Their reduction implies that there is no algorithm with running time  $n^{o(m/\log(m))} \cdot (\Delta + \|b\|_\infty)^{o(m)}$ , since in their construction the matrix  $A$  is non-negative and therefore columns with entries larger than  $\|b\|_\infty$  can be discarded; thus leading to  $\Delta \leq \|b\|_\infty$ . As opposed to our bounds, theirs does not give a precise value for the constant in the exponent.

## 2 Preliminaries

In this paper we are assuming a word size of  $O(m \log(m\Delta) + \log(\|b\|_\infty) + \log(\|c\|_\infty))$  in the word RAM model, that is to say, arithmetic operations on numbers of this encoding size take constant time. When considering  $m$  to be a constant, this makes perfect sense. Also, since we are going to use algorithms with space roughly  $O(m\Delta)^m$ , it is only natural to assume that a single pointer fits into a word.

In the remainder of the paper we will assume that  $A$  has no duplicate columns. Note that we can completely ignore a column  $i$ , if there is another identical column  $i'$  with  $c_{i'} \geq c_i$ . This implies that in time  $O(nm) + O(\Delta)^m$  we can reduce to an instance without duplicate columns and, in particular, with  $n \leq (2\Delta + 1)^m$ . The running time can be achieved as follows.

## 43:4 On Integer Programming and Convolution

We create a new matrix for the ILP with all  $(2\Delta + 1)^m$  possible columns (in lexicographic order) and objective value  $c_i = -\infty$  for all columns  $i$ . Now we iterate over all  $n$  old columns and compute in time  $O(m)$  the index of the new column corresponding to the same entries. We then replace its objective value with the current one if this is bigger. In the upcoming running times we will omit the additive term  $O(nm)$  and assume the duplicates are already eliminated ( $O(\Delta)^m$  is always dominated by actual algorithms running time).

► **Theorem 1 (Steinitz Lemma).** *Let  $\|\cdot\|$  be a norm in  $\mathbb{R}^m$  and let  $v^{(1)}, \dots, v^{(t)} \in \mathbb{R}^m$  such that  $\|v^{(i)}\| \leq 1$  for all  $i$  and  $v^{(1)} + \dots + v^{(t)} = 0$ . Then there exists a permutation  $\pi \in S_t$  such that for all  $j \in \{1, \dots, t\}$*

$$\left\| \sum_{i=1}^j v^{(\pi(i))} \right\| \leq m.$$

The proof for bound  $m$  is due to Sevast'janov [16] (see also [8] for a good overview). Eisenbrand and Weismantel observed that the Steinitz Lemma implies the following.

► **Corollary 2 ([8]).** *Let  $v^{(1)}, \dots, v^{(t)}$  denote columns of  $A$  with  $\sum_{i=1}^t v^{(i)} = b$ . Then there exists a permutation  $\pi \in S_t$  such that for all  $j \in \{1, \dots, t\}$*

$$\left\| \sum_{i=1}^j v^{(\pi(i))} - \frac{j}{t} \cdot b \right\|_{\infty} \leq 2m\Delta.$$

This can be obtained by inserting  $(v^{(i)} - b/t)/(2\Delta)$ ,  $i \in \{1, \dots, t\}$ , in the Steinitz Lemma. Note that  $\|v^{(i)} - b/t\|_{\infty} \leq 2\Delta$ .

► **Lemma 3.** *Let  $\max\{c^T x : Ax = b, x \in \mathbb{Z}_{\geq 0}^n\}$  be bounded and feasible. Then there exists an optimal solution  $x^*$  with  $\|x^*\|_1 \leq O(m\Delta)^m (\|b\|_{\infty} + 1)$ .*

A similar bound is proved for example in [15]. However, we can also give a proof via the Steinitz Lemma.

**Proof.** Let  $x^*$  be an optimal solution of minimal 1-norm. Let  $v^{(1)}, \dots, v^{(t)}$  denote the multiset of columns of  $A$  that represent  $x^*$ . Assume w.l.o.g. these vectors are ordered as in the previous corollary. There cannot be a circle of positive value in  $v^{(1)}, \dots, v^{(t)}$  or else the ILP would be unbounded. By circle we mean a non-empty subset that sums up to 0 and we consider the value of the columns with regard to  $c$ . In fact, there cannot be a circle of nonpositive value either, since the 1-norm of the solution is minimal. Hence, each vector in  $\mathbb{Z}^m$  is visited at most once by the walk  $v^{(1)}, v^{(1)} + v^{(2)}, \dots, v^{(1)} + \dots + v^{(t)} = b$ . The number of integer points  $a$  with

$$\|a - \gamma b\|_{\infty} \leq 2m\Delta \tag{1}$$

for some  $\gamma \in [0, 1]$  is at most  $O(m\Delta)^m \cdot (\|b\|_{\infty} + 1)$  and this upper bounds the 1-norm of  $x^*$ : Assume w.l.o.g.  $\|b\|_{\infty} > 0$  as the case  $b = 0$  is trivial. Take  $\|b\|_{\infty} + 1$  many points evenly distributed along the line from 0 to  $b$ , i.e.,  $b \cdot 0/\|b\|_{\infty}$ ,  $b \cdot 1/\|b\|_{\infty}, \dots, b \cdot \|b\|_{\infty}/\|b\|_{\infty}$ . Then the distance between two consecutive points is small:

$$\left\| b \cdot \frac{j+1}{\|b\|_{\infty}} - b \cdot \frac{j}{\|b\|_{\infty}} \right\|_{\infty} = \left\| \frac{b}{\|b\|_{\infty}} \right\|_{\infty} = 1.$$

In particular, for every vector of the form  $\gamma b$ ,  $\gamma \in [0, 1]$ , there is a point  $b \cdot j/\|b\|_\infty$  that is not further away than  $1/2$ . Thus, for every  $a$  that satisfies (1), we have a point  $b \cdot j/\|b\|_\infty$  with

$$\left\| a - b \cdot \frac{j}{\|b\|_\infty} \right\|_\infty \leq \|a - \gamma b\|_\infty + \left\| \gamma b - b \cdot \frac{j}{\|b\|_\infty} \right\|_\infty \leq 2m\Delta + 1/2.$$

To upper bound the number of vectors of type (1), we count the number of vectors within distance at most  $2m\Delta + 1/2$  to each of the  $\|b\|_\infty + 1$  points. This number is at most  $(\|b\|_\infty + 1) \cdot (4m\Delta + 2)^m$ . This concludes the proof.  $\blacktriangleleft$

**► Corollary 4.** *By adding a zero column we can assume w.l.o.g., if the ILP is feasible and bounded, then there exists an optimal solution  $x^*$  with  $\|x^*\|_1 = U$  where  $U$  is the upper bound for  $\|x^*\|_1$ . By scaling the bound of Lemma 3 to the next power of 2, we can assume that  $\|x^*\|_1 = 2^K$  where  $K \in \mathbb{N}$  and  $K \leq O(m \log(m\Delta) + \log(\|b\|_\infty))$ .*

### 3 Dynamic Program

In this section we will show how to compute the best solution  $x^*$  to an ILP with the additional constraint  $\|x^*\|_1 = 2^K$ . If the ILP is bounded, then with  $K = O(m \log(m\Delta) + \log(\|b\|_\infty))$  and an extra zero column this is the optimum to the ILP (Corollary 4). In Section 3.2 we discuss how to cope with unbounded ILPs. For every  $i = K, K-1, \dots, 0$  and every  $b'$  with

$$\|b' - 2^{-i} \cdot b\|_\infty \leq 4m\Delta$$

we solve  $\max\{c^T x : Ax = b', \|x\|_1 = 2^{K-i}, x \in \mathbb{Z}_{\geq 0}^n\}$ . We start by computing these for  $i = K$  and then iteratively derive solutions for smaller values of  $i$  using only the bigger ones. Ultimately, we will compute a solution for  $i = 0$  and  $b' = b$ .

If  $i = K$ , then every solution must consist of exactly one column ( $\|x\|_1 = 1$ ). We can compute this solution by finding the column that equals  $b'$  should there exist one and set  $-\infty$  otherwise.

Fix some  $i < K$  and  $b'$  and let  $v^{(1)}, \dots, v^{(t)}$  be columns of  $A$  that correspond to an optimal solution to  $\max\{c^T x : Ax = b', \|x\|_1 = 2^{K-i}, x \in \mathbb{Z}_{\geq 0}^n\}$ . In particular,  $v^{(1)} + \dots + v^{(t)} = b'$  and  $t = 2^{K-i}$ . Assume w.l.o.g. that the  $v^{(i)}$  are ordered such that for all  $j \in \{0, \dots, t\}$

$$\left\| \sum_{i=1}^j v^{(i)} - \frac{j}{t} \cdot b' \right\|_\infty \leq 2m\Delta.$$

Note that  $v^{(1)}, \dots, v^{(t/2)}$  is an optimal solution to  $\max\{c^T x : Ax = b'', \|x\|_1 = 2^{K-(i+1)}, x \in \mathbb{Z}_{\geq 0}^n\}$  where  $b'' = v^{(1)} + \dots + v^{(t/2)}$ . Likewise,  $v^{(t/2+1)}, \dots, v^{(t)}$  is an optimal solution to  $\max\{c^T x : Ax = b' - b'', \|x\|_1 = 2^{K-(i+1)}, x \in \mathbb{Z}_{\geq 0}^n\}$ . We claim that  $\|b'' - 2^{-(i+1)} \cdot b\|_\infty \leq 4m\Delta$  and  $\|(b' - b'') - 2^{-(i+1)} \cdot b\|_\infty \leq 4m\Delta$ . This implies that we can look up solutions for  $b''$  and  $b' - b''$  in the dynamic table and their union is a solution for  $b'$ . Clearly it is also optimal. We do not know  $b''$ , but we can guess it: There are only  $(8m\Delta + 1)^m$  candidates. To compute an entry, we therefore enumerate all possible  $b''$  and take the two partial solutions (for  $b''$  and  $b' - b''$ ), where the sum of both values is maximized.



**Proof of claim**

We have that,

$$\begin{aligned} \left\| \sum_{i=1}^{t/2} v^{(i)} - 2^{-(i+1)} \cdot b \right\|_{\infty} &= \left\| \sum_{i=1}^{t/2} v^{(i)} - \frac{1}{2} \cdot b' + \frac{1}{2} \cdot b' - 2^{-(i+1)} \cdot b \right\|_{\infty} \\ &\leq \left\| \sum_{i=1}^{t/2} v^{(i)} - \frac{1}{2} \cdot b' \right\|_{\infty} + \left\| \frac{1}{2} \cdot b' - 2^{-(i+1)} \cdot b \right\|_{\infty} \leq 2m\Delta + \frac{1}{2} \|b' - 2^{-i} \cdot b\|_{\infty} \leq 4 \cdot m\Delta. \end{aligned}$$

In a similar way, we can show that

$$\begin{aligned} \left\| \sum_{i=t/2+1}^t v^{(i)} - 2^{-(i+1)} \cdot b \right\|_{\infty} &= \left\| \sum_{i=t/2+1}^t v^{(i)} - \sum_{i=1}^t v^{(i)} + b' - 2^{-(i+1)} \cdot b \right\|_{\infty} \\ &= \left\| \frac{1}{2} \cdot b' - \sum_{i=1}^{t/2} v^{(i)} + \frac{1}{2} \cdot b' - 2^{-(i+1)} \cdot b \right\|_{\infty} \\ &\leq \left\| \sum_{i=1}^{t/2} v^{(i)} - \frac{1}{2} \cdot b' \right\|_{\infty} + \left\| \frac{1}{2} \cdot b' - 2^{-(i+1)} \cdot b \right\|_{\infty} \leq 4 \cdot m\Delta. \end{aligned}$$

**3.1 Naive running time**

The dynamic table has  $(K+1) \cdot O(m\Delta)^m$  entries. To compute an entry,  $O(n \cdot m) \leq O(m\Delta)^m$  operations are necessary during initialization and  $O(m\Delta)^m$  in the iterative calculations. This gives a total running time of

$$O(m\Delta)^{2m} \cdot (K+1) = O(m\Delta)^{2m} \cdot (m \log(m\Delta) + \log(\|b\|_{\infty})) = O(m\Delta)^{2m} \cdot (\log(\Delta) + \log(\|b\|_{\infty})).$$

Note that  $O(m\Delta)^{2m} = O(m\Delta)^{2m} \cdot 2^m$  hides factors polynomial in  $m$ .

**3.2 Unbounded solutions**

In the previous dynamic program there is no mechanism for detecting when the ILP is unbounded. We follow the approach from [8] to handle unbounded ILPs. The ILP  $\max\{c^T x : Ax = b, x \in \mathbb{Z}_{\geq 0}^n\}$  is unbounded, if and only if  $\{x : Ax = b, x \in \mathbb{Z}_{\geq 0}^n\}$  has a solution and  $\max\{c^T x : Ax = 0, x \in \mathbb{Z}_{\geq 0}^n\}$  has any solution with positive objective value. After running the dynamic program - thereby verifying that there exists any solution - we have to check if the latter condition holds. We can simply run the algorithm again on  $\max\{c^T x : Ax = 0, x \in \mathbb{Z}_{\geq 0}^n\}$  with  $K = m \cdot \lceil \log(2m\Delta + 1) \rceil$ . If it returns a positive value, the ILP is unbounded. Let us argue why this is enough. We need to understand that when there is a positive solution to  $\max\{c^T x : Ax = 0, x \in \mathbb{Z}_{\geq 0}^n\}$ , then there is also a positive solution with 1-norm at most  $(2m\Delta + 1)^m \leq 2^K$ . Let  $x^*$  be a positive solution to the former ILP with minimal 1-norm, i.e.,  $c^T x^* > 0$  and  $\|x^*\|_1$  minimal. Let  $v^{(1)}, \dots, v^{(t)}$  be the multiset of columns representing  $x^*$ . We assume that they are ordered as in Corollary 2. If  $t > (2m\Delta + 1)^m$ , then there must be two identical partial sums  $\sum_{i=1}^j v^{(i)} = \sum_{i=1}^k v^{(i)}$  with  $j < k$ . In other words, the circle can be decomposed into two circles  $v^{(1)}, \dots, v^{(j)}, v^{(k+1)}, \dots, v^{(t)}$  and  $v^{(j+1)}, \dots, v^{(k)}$ . One of these must be a positive solution or else their sum would be negative. This means the 1-norm of  $x^*$  is not minimal. We conclude that  $t \leq (2m\Delta + 1)^m$ .

## 4 Improvements to the running time

### 4.1 Applying convolution

Can we speed up the computation of entries in the dynamic table? Let  $D_i$  be the set of vectors  $b'$  with  $\|b' - 2^{-i} \cdot b\|_\infty \leq 4m\Delta$ . Recall, the dynamic programs computes values for each element in  $D_K, D_{K-1}, \dots, D_1$ . More precisely for the value of  $b' \in D_i$  we consider vectors  $b''$  such that  $b'', b' - b'' \in D_{i+1}$  and take the maximum sum of the values for  $b'', b' - b''$  among all. First consider only the case of  $m = 1$ . Here we have that  $b' \in D_i$  is equivalent to  $-4\Delta \leq b' - 2^{-i} \cdot b \leq 4\Delta$ . This problem is well studied. It is a variant of (min, +)-convolution.

(MIN, +)-CONVOLUTION

**Input:**  $r_1, \dots, r_n$  and  $s_1, \dots, s_n$ .

**Output:**  $t_1, \dots, t_n$ , where  $t_k = \min_{i+j=k} r_i + s_j$ .

(max, +)-convolution is the counterpart where the maximum is taken instead of the minimum. The two problems are equivalent. Each of them can be transformed to the other by negating the elements. We construct an instance of (max, +)-convolution of size  $12\Delta + 2$ . We set  $r_j$  and  $s_j$ ,  $j \in \{1, \dots, 8\Delta + 1\}$  both to the value for  $b/2^{i+1} - (4\Delta + 1) + j \in D_{i+1}$  in the dynamic table. Set the remaining values of  $r$  and  $s$  to  $-\infty$ . Then for  $b' = b/2^i - (4\Delta + 1) + k \in D_i$ , the correct result will be at  $t_{4\Delta+1+k}$ .

(min, +)-convolution admits a trivial  $O(n^2)$  time algorithm and it has been conjectured that there exists no truly sub-quadratic algorithm [7]. There does, however, exist an  $O(n^2 / \log(n))$  time algorithm [4], which we are going to use. In fact, there is a slightly faster algorithm with running time  $n^2 / 2^{\Omega(\sqrt{\log(n)})}$  [6].

We can reduce the problem for arbitrary  $m$  to a (max, +)-convolution instance of size  $O(m\Delta)^m$ . To do so, project a vector  $b' \in D_i$  to

$$f_i(b') = \sum_{j=1}^m (16m\Delta + 3)^{j-1} \underbrace{(4m\Delta + 1 + b'_j - b_j/2^i)}_{\in [1, 8m\Delta+1]}. \quad (2)$$

The value  $16m\Delta + 3$  is chosen because it is always greater than the sum of two values of the form  $4m\Delta + 1 + b'_j - b_j/2^i$ . For all  $a, a' \in D_{i+1}, b' \in D_i$ , it holds that  $f_{i+1}(a) + f_{i+1}(a') = f_i(b')$ , if and only if  $a + a' = b' - (4m\Delta + 1, \dots, 4m\Delta + 1)^T$ :

**Proof**  $\Rightarrow$ . Let  $f_{i+1}(a) + f_{i+1}(a') = f_i(b')$ . Then in particular,

$$f_{i+1}(a) + f_{i+1}(a') \equiv f_i(b') \pmod{16m\Delta + 3}$$

Since all but the first element of the sum (2) are multiples of  $16m\Delta + 3$ , i.e., they are equal 0 modulo  $16m\Delta + 3$ , we can omit them in the equation. Hence,

$$(4m\Delta + 1 + a_1 - b_1/2^{i+1}) + (4m\Delta + 1 + a'_1 - b_1/2^{i+1}) \equiv (4m\Delta + 1 + b'_1 - b_1/2^i) \pmod{16m\Delta + 3}.$$

We even have equality (without modulo) here, because both sides are smaller than  $16m\Delta + 3$ . Simplifying the equation gives  $a_1 + a'_1 = b'_1 - (4m\Delta + 1)$ . Now consider again the equation  $f_{i+1}(a) + f_{i+1}(a') = f_i(b')$ . In the sums leave out the first element. The equation still holds, since by the elaboration above this changes the left and right hand-side by the same value. We can now repeat the same argument to obtain  $a_2 + a'_2 = b'_2 - (4m\Delta + 1)$  and the same for all other dimensions.  $\blacktriangleleft$

**Proof**  $\Leftarrow$ . Let  $a + a' = b' - (4m\Delta + 1, \dots, 4m\Delta + 1)^T$ . Then for every  $j$ ,

$$(4m\Delta + 1 + a_j - b_j/2^{i+1}) + (4m\Delta + 1 + a'_j - b_j/2^{i+1}) = 4m\Delta + 1 + b'_j - b_j/2^i.$$

It directly follows that  $f_{i+1}(a) + f_{i+1}(a') = f_i(b')$ .  $\blacktriangleleft$

This means when we write the value of each  $b'' \in D_{i+1}$  to  $r_j$  and  $s_j$ , where  $j = f_{i+1}(b'')$ , the correct solutions will be in  $t$ . More precisely, we can read the result for some  $b' \in D_i$  at  $t_k$  where  $k = f_i(b' + (4m\Delta + 1, \dots, 4m\Delta + 1)^T)$ .

With an algorithm for  $(\min, +)$ -convolution with running time  $T(n)$  we get an algorithm with running time  $T(O(m\Delta)^m) \cdot (m \log(m\Delta) + \log(\|b\|_\infty))$ . Inserting  $T(n) = n^2/\log(n)$  we get:

**► Theorem 5.** *There exists an algorithm that finds the optimum of  $\max\{c^T x : Ax = b, x \in \mathbb{Z}_{\geq 0}^n\}$ , in time  $O(m\Delta)^{2m} \cdot (1 + \log(\|b\|_\infty)/\log(\Delta))$ .*

Clearly, a sub-quadratic algorithm, where  $T(n) = n^{2-\delta}$  for some  $\delta > 0$ , would directly improve the exponent. Next, we will consider the problem of only testing feasibility of an ILP. Since we only record whether or not there exists a solution for a particular right-hand side, the convolution problem reduces to the following.

**BOOLEAN CONVOLUTION**  
**Input:**  $r_1, \dots, r_n \in \{0, 1\}$  and  $s_1, \dots, s_n \in \{0, 1\}$ .  
**Output:**  $t_1, \dots, t_n \in \{0, 1\}$ , where  $t_k = \bigvee_{i+j=k} r_i \wedge s_j$ .

This problem can be solved very efficiently via fast Fourier transform. We compute the  $(+, \cdot)$ -convolution of the input. It is well known that this can be done using FFT in time  $O(n \log(n))$ . The  $(+, \cdot)$ -convolution of  $r$  and  $s$  is the vector  $t$ , where  $t_k = \sum_{i+j=k} r_i \cdot s_j$ . To get the Boolean convolution instead, we simply replace each  $t_k > 0$  by 1. Using  $T(n) = O(n \log(n))$  for the convolution algorithm we obtain the following.

**► Theorem 6.** *There exists an algorithm that finds an element in  $\{x : Ax = b, x \in \mathbb{Z}_{\geq 0}^n\}$ , if there is one, in time  $O(m\Delta)^m \cdot \log(\Delta) \cdot \log(\Delta + \|b\|_\infty)$ .*

This can be seen from the calculation below. First we scrape off factors polynomial in  $m$ :

$$O(m\Delta)^m \cdot m \log(m\Delta) \cdot (m \log(m\Delta) + \log(\|b\|_\infty)) \leq O(m\Delta)^m \cdot \log(\Delta) \cdot (\log(\Delta) + \log(\|b\|_\infty))$$

Next, we use that  $\log(\Delta) + \log(\|b\|_\infty) = \log(\Delta \cdot \|b\|_\infty) \leq \log((\Delta + \|b\|_\infty)^2) = O(\log(\Delta + \|b\|_\infty))$ .

## 4.2 Use of proximity

Eisenbrand and Weismantel gave the following bound on the proximity between continuous and integral solutions.

**► Theorem 7 ([8]).** *Let  $\max\{c^T x : Ax = b, x \in \mathbb{Z}_{\geq 0}^n\}$  be feasible and bounded. Let  $x^*$  be an optimal vertex solution of the fractional relaxation. Then there exists an optimal solution  $z^*$  with*

$$\|z^* - x^*\|_1 \leq m(2m\Delta + 1)^m.$$

We briefly explain, how they use this theorem to reduce the right-hand side  $b$  at the expense of computing the optimum of the fractional relaxation: Note that  $z_i^* \geq \ell_i := \max\{0, \lceil x_i^* \rceil - m(2m\Delta + 1)^m\}$ . Since  $x^*$  is a vertex solution, it has at most  $m$  non-zero components. By

setting  $y = x - \ell$  we obtain the equivalent ILP  $\max\{c^T y : Ay = b - A\ell, y \in \mathbb{Z}_{\geq 0}^n\}$ . Indeed, this ILP has a bounded right-hand side:

$$\|b - A\ell\|_{\infty} = \|A(x^* - \ell)\|_{\infty} \leq \Delta m^2 (2m\Delta + 1)^m = O(m\Delta)^{m+1}.$$

Here, we use that  $x^*$  and  $\ell$  differ only in non-zero components of  $x^*$  and in those by at most  $m(2m\Delta + 1)^m$ . Like in earlier bounds, the O-notation hides polynomial terms in  $m$ . Using the  $n \cdot O(m\Delta)^{2m} \cdot \|b\|_1^2$  time algorithm from [8], this gives a running time of  $n \cdot O(m\Delta)^{4m+2} + \text{LP}$ , where LP is the time to solve the relaxation. The logarithmic dependence on  $\|b\|_{\infty}$  in our new algorithm leads to a much smaller exponent: Using Theorem 5 and the construction above, the ILP can be solved in time  $O(m\Delta)^{2m} + \text{LP}$ . Feasibility can be tested in time  $O(m\Delta)^m \cdot \log^2(\Delta) + \text{LP}$  using Theorem 6.

### 4.3 Heterogeneous matrices

Let  $\Delta_1, \dots, \Delta_m \leq \Delta$  denote the largest absolute values of each row in  $A$ . When some of these values are much smaller than  $\Delta$ , the maximum among all, we can do better than  $O(m\Delta)^{2m} \cdot \log(\|b\|_{\infty})$ . An example for a highly heterogeneous matrix is UNBOUNDED KNAPSACK with cardinality constraints. Consider the norm  $\|v\| = 1/2 \cdot \max_k |v_k/\Delta_k|$  and let  $v^{(1)}, \dots, v^{(t)} \in \mathbb{Z}^m$  be the multiset of columns corresponding to an optimal solution of the ILP. Using the Steinitz Lemma on this norm, it follows that there exists a permutation  $\pi$  such that for all  $j \in \{1, \dots, t\}$  and  $k \in \{1, \dots, k\}$

$$\left| \sum_{i=1}^j v_k^{(\pi(i))} - \frac{j}{t} \cdot b_k \right| \leq 2m\Delta_k.$$

This means the number of states we have to consider reduces from  $O(m\Delta)^m$  to  $\prod_{k=1}^m O(m\Delta_k)$  at each level of the dynamic program. Hence, we obtain the running time  $\prod_{k=1}^m O(m\Delta_k)^2 \cdot \log(\|b\|_{\infty})$ . When the objective function has small coefficients, it is more efficient to perform a binary search for the optimum and encode the objective function as an additional constraint. We can bound the optimum by  $O(m\Delta)^m \cdot (\|b\|_{\infty} + 1) \cdot \|c\|_{\infty}$  using the bound on the 1-norm of the solution. Hence, the binary search takes at most  $O(m \log(m\Delta \cdot \|c\|_{\infty} \cdot \|b\|_{\infty})) = O(m \log(m\Delta + \|c\|_{\infty} + \|b\|_{\infty}))$  iterations. For a guess  $\tau$  the following feasibility ILP tests if there is a solution of value at least  $\tau$ .

$$\begin{pmatrix} c_1 & \dots & c_n & -1 \\ & & & 0 \\ & A & & \vdots \\ & & & 0 \end{pmatrix} x = \begin{pmatrix} \tau \\ b_1 \\ \vdots \\ b_n \end{pmatrix}$$

$$x \in \mathbb{Z}_{\geq 0}^{n+1}$$

We can solve the ILP above in time

$$T(\|c\|_{\infty} \cdot \prod_{k=1}^m O((m+1)\Delta_k)) \cdot \log(\|b\|_{\infty} + \tau) \leq T(\|c\|_{\infty} \cdot \prod_{k=1}^m O(m\Delta_k)) \cdot m \log(m\Delta + \|c\|_{\infty} + \|b\|_{\infty}),$$

where  $T(n) = O(n \log(n))$  is the running time of Boolean convolution. By adding the time for the binary search and by hiding polynomials in  $m$ , we get the total running time of

$$\|c\|_{\infty} \cdot \prod_{k=1}^m [O(m\Delta_k)] \cdot \log(\Delta + \|c\|_{\infty}) \cdot \log^2(\Delta + \|c\|_{\infty} + \|b\|_{\infty}).$$

**5 Lower bounds**

**5.1 Optimization problem**

We use an equivalence between UNBOUNDED KNAPSACK and (min, +)-convolution regarding sub-quadratic algorithms.

UNBOUNDED KNAPSACK  
**Input:**  $C \in \mathbb{N}$ ,  $w_1, \dots, w_n \in \mathbb{N}$ , and  $p_1, \dots, p_n \in \mathbb{N}$ .  
**Output:** Multiplicities  $x_1, \dots, x_n$ , such that  $\sum_{i=1}^n x_i \cdot w_i \leq C$  and  $\sum_{i=1}^n x_i \cdot p_i$  is maximized.

Note that when we instead require  $\sum_{i=1}^n x_i \cdot w_i = C$  in the problem above, we can transform it to this form by adding an item of profit zero and weight 1.

► **Theorem 8** ([7]). *For any  $\delta > 0$  there exists no  $O((n + C)^{2-\delta})$  time algorithm for UNBOUNDED KNAPSACK unless there exists a truly sub-quadratic algorithm for (min, +)-convolution.*

When using this theorem, we assume that the input already consists of the at most  $C$  relevant items only,  $n \leq C$ , and  $w_i \leq C$  for all  $i$ . This preprocessing can be done in time  $O(n + C)$ .

► **Theorem 9.** *For every fixed  $m$  there does not exist an algorithm that solves ILPs with  $m$  constraints in time  $f(m) \cdot (n^{2-\delta} + (\Delta + \|b\|_\infty)^{2m-\delta})$  for some  $\delta > 0$  and a computable function  $f$ , unless there exists a truly sub-quadratic algorithm for (min, +)-convolution.*

**Proof.** Let  $\delta > 0$  and  $m \in \mathbb{N}$ . Assume that there exists an algorithm that solves ILPs of the form  $\max\{c^T x : Ax = b, x \in \mathbb{Z}_{\geq 0}^n\}$  where  $A \in \mathbb{Z}^{m \times n}$ ,  $b \in \mathbb{Z}^m$ , and  $c \in \mathbb{Z}^n$  in time  $f(m) \cdot (n^{2-\delta} + (\Delta + \|b\|_\infty)^{2m-\delta})$ , where  $\Delta$  is the greatest absolute value in  $A$ . We will show that this implies an  $O((n + C)^{2-\delta'})$  time algorithm for the UNBOUNDED KNAPSACK PROBLEM for some  $\delta' > 0$ . Let  $(C, (w_i)_{i=1}^n, (p_i)_{i=1}^n)$  be an instance of this problem. Let us first observe that the claim holds for  $m = 1$ . Clearly the UNBOUNDED KNAPSACK PROBLEM (with equality) can be written as the following ILP (UKS1).

$$\begin{aligned} & \max \sum_{i=1}^n p_i \cdot x_i \\ & \sum_{i=1}^n w_i \cdot x_i = C \\ & x \in \mathbb{Z}_{\geq 0}^n \end{aligned}$$

Since  $w_i \leq C$  for all  $i$  (otherwise the item can be discarded), we can solve this ILP by assumption in time  $f(1) \cdot (n^{2-\delta} + (2C)^{2-\delta}) \leq O((n + C)^{2-\delta})$ . Now consider the case where  $m > 1$ . We want to reduce  $\Delta$  by exploiting the additional rows. Let  $\Delta = \lfloor C^{1/m} \rfloor + 1 > C^{1/m}$ . We write  $C$  in base- $\Delta$  notation, i.e.,

$$C = C^{(0)} + \Delta C^{(1)} + \dots + \Delta^{m-1} C^{(m-1)},$$

where  $0 \leq C^{(k)} < \Delta$  for all  $k$ . Likewise, write  $w_i = w_i^{(0)} + \Delta w_i^{(1)} + \dots + \Delta^{m-1} w_i^{(m-1)}$  with

$0 \leq w_i^{(k)} < \Delta$  for all  $k$ . We claim that (UKS1) is equivalent to the following ILP (UKSm).

$$\begin{aligned} \max \sum_{i=1}^n p_i \cdot x_i \\ \sum_{i=1}^n [w_i^{(0)} \cdot x_i] - \Delta \cdot y_1 = C^{(0)} \end{aligned} \quad (3)$$

$$\sum_{i=1}^n [w_i^{(1)} \cdot x_i] + y_1 - \Delta \cdot y_2 = C^{(1)} \quad (4)$$

$\vdots$

$$\sum_{i=1}^n [w_i^{(m-2)} \cdot x_i] + y_{m-2} - \Delta \cdot y_{m-1} = C^{(m-2)} \quad (5)$$

$$\sum_{i=1}^n [w_i^{(m-1)} \cdot x_i] + y_{m-1} = C^{(m-1)} \quad (6)$$

$$\begin{aligned} x &\in \mathbb{Z}_{\geq 0}^n \\ y &\in \mathbb{Z}_{\geq 0}^m \end{aligned}$$

**Claim**  $x \in (\text{USK1}) \Rightarrow x \in (\text{USKm})$

Let  $x$  be a solution to (UKS1). Then for all  $1 \leq \ell \leq m$ ,

$$\sum_{i=1}^n \sum_{k=0}^{\ell-1} \Delta^k w_i^{(k)} \cdot x_i \equiv \sum_{i=1}^n w_i \cdot x_i \equiv C \equiv \sum_{k=0}^{\ell-1} \Delta^k C^{(k)} \pmod{\Delta^\ell}.$$

This is because all  $\Delta^\ell w_i^{(\ell)}, \dots, \Delta^{m-1} w_i^{(m-1)}$  and  $\Delta^\ell C^{(\ell)}, \dots, \Delta^{m-1} C^{(m-1)}$  are multiples of  $\Delta^\ell$ . It follows that there exists an  $y_\ell \in \mathbb{Z}$  such that

$$\sum_{i=1}^n \sum_{k=0}^{\ell-1} [\Delta^k w_i^{(k)} \cdot x_i] - \Delta^\ell \cdot y_\ell = \sum_{k=0}^{\ell-1} \Delta^k C^{(k)}.$$

Furthermore,  $y_\ell$  is non-negative, because otherwise

$$\begin{aligned} \sum_{k=0}^{\ell-1} \Delta^k C^{(k)} &\leq \sum_{k=0}^{\ell-1} \Delta^k (\Delta - 1) < \Delta^{\ell-1} (\Delta - 1) \sum_{k=0}^{\infty} \Delta^{-k} \\ &= \Delta^{\ell-1} \frac{\Delta - 1}{1 - \frac{1}{\Delta}} = \Delta^\ell \leq -\Delta^\ell y_\ell \leq \sum_{i=1}^n \sum_{k=0}^{\ell-1} [\Delta^k w_i^{(k)} \cdot x_i] - \Delta^\ell y_\ell. \end{aligned}$$

We choose  $y_1, \dots, y_m$  exactly like this. The first constraint (3) follows directly. Now let  $\ell \in \{2, \dots, m\}$ . By choice of  $y_{\ell-1}$  and  $y_\ell$  we have that

$$\sum_{i=1}^n \left[ \underbrace{\left( \sum_{k=0}^{\ell-1} \Delta^k w_i^{(k)} - \sum_{k=0}^{\ell-2} \Delta^k w_i^{(k)} \right)}_{=\Delta^{\ell-1} w_i^{(\ell-1)}} \cdot x_i \right] + \Delta^{\ell-1} \cdot y_{\ell-1} - \Delta^\ell \cdot y_\ell = \underbrace{\sum_{k=0}^{\ell-1} \Delta^k C^{(k)} - \sum_{k=0}^{\ell-2} \Delta^k C^{(k)}}_{=\Delta^{\ell-1} C^{(\ell-1)}}. \quad (7)$$

## 43:12 On Integer Programming and Convolution

Dividing both sides by  $\Delta^{\ell-1}$  we get every constraint (4) - (5) for the correct choice of  $\ell$ . Finally, consider the special case of the last constraint (6). By choice of  $y_m$  we have that

$$\sum_{i=1}^n \underbrace{\sum_{k=0}^{m-1} \Delta^k w_i^{(k)}}_{=w_i} \cdot x_i - \Delta^m \cdot y_m = \underbrace{\sum_{k=0}^{m-1} \Delta^k C^{(k)}}_{=C}.$$

Thus,  $y_m = 0$  and (7) implies the last constraint (with  $\ell = m$ ).

**Claim**  $x \in (\text{USK}m) \Rightarrow x \in (\text{USK}1)$

Let  $x_1, \dots, x_n, y_1, \dots, y_{m-1}$  be a solution to (UKSm) and set  $y_m = 0$ . We show by induction that for all  $\ell \in \{1, \dots, m\}$

$$\sum_{i=1}^n \sum_{k=0}^{\ell-1} \Delta^k w_i^{(k)} \cdot x_i - \Delta^\ell y_\ell = \sum_{k=0}^{\ell-1} \Delta^k C^{(k)}.$$

With  $\ell = m$  this implies the claim as  $y_m = 0$  by definition. For  $\ell = 1$  the equation is exactly the first constraint (3). Now let  $\ell > 1$  and assume that the equation above holds. We will show that it also holds for  $\ell + 1$ . From (USK $m$ ) we have

$$\sum_{i=1}^n [w_i^{(\ell)} \cdot x_i] + y_\ell - \Delta \cdot y_{\ell+1} = C^{(\ell)}.$$

Multiplying each side by  $\Delta^\ell$  we get

$$\sum_{i=1}^n [\Delta^\ell w_i^{(\ell)} \cdot x_i] + \Delta^\ell y_\ell - \Delta^{\ell+1} \cdot y_{\ell+1} = \Delta^\ell C^{(\ell)}.$$

By adding and subtracting the same elements, it follows that

$$\sum_{i=1}^n \left[ \left( \sum_{k=0}^{\ell} \Delta^k w_i^{(k)} - \sum_{k=0}^{\ell-1} \Delta^k w_i^{(k)} \right) \cdot x_i \right] + \Delta^\ell \cdot y_\ell - \Delta^{\ell+1} \cdot y_{\ell+1} = \sum_{k=0}^{\ell} \Delta^k C^{(k)} - \sum_{k=0}^{\ell-1} \Delta^k C^{(k)}.$$

By inserting the induction hypothesis we conclude

$$\sum_{i=1}^n \sum_{k=0}^{\ell} [\Delta^k w_i^{(k)} \cdot x_i] - \Delta^{\ell+1} y_{\ell+1} = \sum_{k=0}^{\ell} \Delta^k C^{(k)}.$$

### Constructing and solving the ILP

The ILP (UKSm) can be constructed easily in  $O(Cm + nm) \leq O((n + C)^{2-\delta/m})$  operations (recall that  $m$  is a constant). We obtain  $\Delta = \lfloor C^{1/m} \rfloor + 1$  by guessing: More precisely, we iterate over all numbers  $\Delta_0 \leq C$  and find the one where  $(\Delta_0 - 1)^m < C \leq \Delta_0^m$ . There are of course more efficient, non-trivial ways to compute the rounded  $m$ -th root. The base- $\Delta$  representation for  $w_1, \dots, w_n$  and  $C$  can be computed with  $O(m)$  operations for each of these numbers.

All entries of the matrix in (UKSm) and the right-hand side are bounded by  $\Delta = O(C^{1/m})$ . Therefore, by assumption this ILP can be solved in time

$$f(m) \cdot (n^{2-\delta} + O(C^{1/m})^{2m-\delta}) \leq f(m) \cdot O(1)^{2m-\delta} \cdot (n + C)^{2-\delta/m} = O((n + C)^{2-\delta/m})$$

This would therefore yield a truly sub-quadratic algorithm for the UNBOUNDED KNAPSACK PROBLEM.  $\blacktriangleleft$



## 5.2 Feasibility problem

We will show that our algorithm for solving feasibility of ILPs is optimal (except for log factors). We use a recently discovered lower bound for k-SUM based on the SETH.

k-SUM

**Input:**  $T \in \mathbb{N}_0$  and  $Z_1, \dots, Z_k \subset \mathbb{N}_0$  where  $|Z_1| + |Z_2| + \dots + |Z_k| = n \in \mathbb{N}$ .

**Output:**  $z_1 \in Z_1, z_2 \in Z_2, \dots, z_k \in Z_k$  such that  $z_1 + z_2 + \dots + z_k = T$ .

► **Theorem 10** ([1]). *If the SETH holds, then for every  $\delta > 0$  there exists a value  $\gamma > 0$  such that k-SUM cannot be solved in time  $O(T^{1-\delta} \cdot n^{\gamma k})$ .*

This implies that for every  $p \in \mathbb{N}$  there is no  $O(T^{1-\delta} \cdot n^p)$  time algorithm for k-SUM if  $k \geq p/\gamma$ .

► **Theorem 11.** *If the SETH holds, for every fixed  $m$  there does not exist an algorithm that solves feasibility of ILPs with  $m$  constraints in time  $n^{f(m)} \cdot (\Delta + \|b\|_\infty)^{m-\delta}$ .*

**Proof.** Like in the previous reduction we start with the case of  $m = 1$ . For higher values of  $m$  the result can be shown in the same way as before.

Suppose there exists an algorithm for solving feasibility of ILPs with one constraint in time  $n^{f(1)} \cdot (\Delta + \|b\|_\infty)^{1-\delta}$  for some  $\delta > 0$  and  $f(1) \in \mathbb{N}$ . Set  $k = \lceil f(1)/\gamma \rceil$  with  $\gamma$  as in in Theorem 10 and consider an instance  $(T, Z_1, \dots, Z_k)$  of k-SUM. We will show that this can be solved in time  $O(T^{1-\delta} \cdot n^{f(1)})$ , which contradicts the SETH. For every  $i \leq k$  and every  $z \in Z_i$  we use a binary variable  $x_{i,z}$  that describes whether  $z$  is used. We can easily model k-SUM as the following ILP:

$$\begin{aligned} \sum_{i=1}^k \sum_{z \in Z_i} z \cdot x_{i,z} &= T \\ \sum_{z \in Z_i} x_{i,z} &= 1 && \forall i \in \{1, \dots, k\} \\ x_{i,z} &\in \mathbb{Z}_{\geq 0} && \forall i \in \{1, \dots, k\}, z \in Z_i \end{aligned}$$

However, since we want to reduce to an ILP with one constraint, we need a slightly more sophisticated construction. We will show that the cardinality constraints can be encoded into the k-SUM instance by increasing the numbers by a factor of  $2^{O(k)}$ , which is in  $O(1)$  since  $k$  is some constant depending on  $f(1)$  and  $\gamma$  only. We will use this to obtain an ILP with only one constraint and values of size at most  $O(T)$ . A similar construction is also used in [1].

Our goal is to construct an instance  $(T', Z'_1, \dots, Z'_k)$  such that for every  $x^*$  it holds that  $x^*$  is a solution to the first ILP if and only if  $x^* \in \{x : \sum_{i=1}^k \sum_{z \in Z'_i} z \cdot x_{i,z} = T', x \in \mathbb{Z}_{\geq 0}^n\}$  (\*). We will use one element to represent each element in the original instance. Consider the binary representation of numbers in  $Z'_1 \cup \dots \cup Z'_k$  and of  $T'$ . The numbers in the new instance will consist of three parts and  $\lceil \log(k) \rceil$  many 0s between them to prevent interference. For an illustration of the construction see Figure 2. The  $\lceil \log(k) \rceil$  most significant bits ensure that exactly  $k$  elements are selected; the middle part are  $k$  bits that ensure of every set  $Z'_i$  exactly one element is selected; the least significant  $\lceil \log(T) \rceil$  bits represent the original values of the elements. Set the values in the first part of the numbers to 1 for all elements  $Z'_1 \cup \dots \cup Z'_k$  and to  $k$  in  $T'$ . Clearly this ensures that at most  $k$  elements are chosen. The sum of at most  $k$  elements cannot be larger than  $k \leq 2^{\lceil \log(k) \rceil}$  times the biggest element. This implies that the buffers of  $\lceil \log(k) \rceil$  zeroes cannot overflow and we can consider each of the three parts independently. It follows that exactly  $k$  elements must be chosen by any

$$\begin{aligned}
 Z'_i \ni z' &= \overbrace{0 \dots 0001}^{\text{bin}(1)} \mid \overbrace{0 \dots 0}^{\lceil \log(k) \rceil} \mid \overbrace{0 \dots 010 \dots 0}^{\text{bin}(2^i)} \mid \overbrace{0 \dots 0}^{\lceil \log(k) \rceil} \mid \overbrace{0110 \dots}^{\text{bin}(z)} \\
 T' &= \overbrace{0 \dots 1011}^{\text{bin}(k)} \mid \overbrace{0 \dots 0}^{\lceil \log(k) \rceil} \mid \overbrace{1111 \dots 1111}^{\text{bin}(2^{k+1}-1)} \mid \overbrace{0 \dots 0}^{\lceil \log(k) \rceil} \mid \overbrace{1011 \dots}^{\text{bin}(T)}
 \end{aligned}$$

■ **Figure 2** Construction of  $Z'_i$  and  $T'$ .

feasible solution. The system  $\{x : \sum_{i=1}^k 2^i x_i = 2^{k+1} - 1, \|x\|_1 = k, \mathbb{Z}_{\geq 0}^k\}$  has exactly one solution and this solution is  $(1, 1, \dots, 1)$ : Consider summing up  $k$  powers of 2 and envision the binary representation of the partial sums. When we add some  $2^i$  to the partial sum, the number of ones in the binary representation increases by one, if the  $i$ 'th bit of the current sum is zero. Otherwise, it does not increase. However, since in the binary representation of the final sum there are  $k$  ones, it has to increase in each addition. This means no power of two can be added twice and therefore each has to be added exactly once.

It follows that the second part of the numbers enforces that of every  $Z'_i$  exactly one element is chosen. We conclude that  $(*)$  solves the initial  $k$ -SUM instance. By assumption this can be done in time  $n^{f(1)} \cdot (\Delta + \|b\|_\infty)^{1-\delta} = n^{f(1)} \cdot O(T')^{1-\delta} = O(n^{f(1)} \cdot T^{1-\delta})$ . Here we use that  $T' \leq 2^{3 \log(k) + k + \log(T) + 4} = O(k^3 2^k T) = O(T)$ , since  $k$  is a constant.

For  $m > 1$  we can use the same construction as in the reduction for the optimization problem: Suppose there is an algorithm that finds feasible solutions to ILPs with  $m$  constraints in time  $n^{f(m)} \cdot (\Delta + \|b\|_\infty)^{m-\delta}$ . Choose  $\gamma$  such that there is no algorithm for  $k$ -SUM with running time  $O(T^{1-\delta/m} \cdot n^{\gamma k})$  (under SETH). We set  $k = \lceil f(m)/\gamma \rceil$ . By splitting the one constraint of  $(*)$  into  $m$  constraints we can reduce the upper bound on elements from  $O(T)$  to  $O(T^{1/m})$ . This means the assumed running time for solving ILPs can be used to solve  $k$ -SUM in time

$$n^{f(m)} \cdot O(T^{1/m})^{m-\delta} \leq n^{\gamma k} \cdot O(1)^{m-\delta} \cdot T^{1-\delta/m} = O(n^{\gamma k} \cdot T^{1-\delta/m}). \quad \blacktriangleleft$$

## 6 Applications

We describe the implications of our results on a couple of well-known problems, which can be formulated using ILPs with few constraints and small entries. In particular, we give an example, where the reduction of the running time by a factor  $n$  improves on the state-of-the-art and one where the logarithmic dependence on  $\|b\|_\infty$  proves useful.

### 6.1 Unbounded Knapsack and Unbounded Subset-Sum

UNBOUNDED KNAPSACK with equality constraint is simply an ILP with  $m = 1$  and positive entries and objective function:

$$\max \left\{ \sum_{i=1}^n p_i \cdot x_i : \sum_{i=1}^n w_i \cdot x_i = C, x \in \mathbb{Z}_{\geq 0}^n \right\}$$

where  $p_i \geq 0$  are called the profits and  $w_i \geq 0$  the weights of the items  $1, \dots, n$ . More common is to let  $C$  be only an upper bound on  $\sum_{i=1}^n w_i \cdot x_i$ , but that variant easily reduces to the problem above by adding a slack variable. UNBOUNDED SUBSET-SUM is the same problem

without an objective function, i.e., the problem of finding a multiset of items whose weights sum up to exactly  $C$ . We assume that no two items have the same weight. Otherwise in time  $O(n + \Delta)$  we can remove all duplicates by keeping only the most valuable ones. The fractional solutions to both problems are of a very simple structure: For UNBOUNDED KNAPSACK choose only the item  $i$  of maximal efficiency, that is  $p_i/w_i$ , and select it  $C/w_i$  times. For UNBOUNDED SUBSET-SUM choose an arbitrary item. This gives algorithms with running time  $O(\Delta^2)$  and  $O(\Delta \log^2(\Delta))$  for UNBOUNDED KNAPSACK and UNBOUNDED SUBSET-SUM, respectively, where  $\Delta$  is the maximum weight among all items (using the results from Section 4.2). The previously best pseudo-polynomial algorithms for UNBOUNDED KNAPSACK, have running times  $O(nC)$  (standard dynamic programming; see e.g. [13]),  $O(n\Delta^2)$  [8], or very recently  $O(\Delta^2 \log(C))$  [2]. We note that the algorithm from the last paper, which was discovered independently and concurrently to our results, also uses (min, +)-convolution. It could probably be improved to the same running time as our general algorithm using the proximity ideas. For UNBOUNDED SUBSET-SUM the state-of-the-art algorithm has a running time  $O(C \log(C))$  [5]. Hence, our algorithm is preferable when  $\Delta \ll C$ .

## 6.2 Scheduling on Identical Machines

The problem SCHEDULING ON IDENTICAL MACHINES asks for the distribution of  $N$  jobs onto  $M \leq N$  machines. Each job  $j$  has a processing time  $p_j$  and the objective is to minimize the makespan, i.e., the maximum sum of processing times on a single machine. Since an exact solution cannot be computed unless  $P = NP$ , we are satisfied with a  $(1 + \epsilon)$ -approximation, where  $\epsilon > 0$  is part of the input. We will outline how this problem can be solved using our algorithm. More details on many of the techniques involved can be found in [10].

We consider here the variant, in which a makespan  $\tau$  is given and we have to find a schedule with makespan at most  $(1 + \epsilon)\tau$  or prove that there exists no schedule with makespan at most  $\tau$ . This suffices by using a standard dual approximation framework. It is easy to see that one can discard all jobs of size at most  $\epsilon \cdot \tau$  and add them greedily after a solution for the other jobs is found. The big jobs can each be rounded to the next value of the form  $\epsilon \cdot \tau \cdot (1 + \epsilon)^i$  for some  $i$ . This reduces the number of different processing times to  $O(1/\epsilon \log(1/\epsilon))$  many and increases the makespan by at most a factor of  $1 + \epsilon$ . We are now ready to write this problem as an ILP. A configuration is a way to use a machine. It describes how many jobs of each size are assigned to this machine. Since we aim for a makespan of  $(1 + \epsilon) \cdot \tau$ , the sum of these sizes must not exceed this value. The configuration ILP has a variable for every valid configuration and it describes how many machines use this configuration. Let  $\mathcal{C}$  be the set of valid configurations and  $C_k$  the multiplicity of size  $k$  in a configuration  $C \in \mathcal{C}$ . The following ILP solves the rounded instance. We note that there is no objective function in it.

$$\begin{aligned} \sum_{C \in \mathcal{C}} x_C &= M \\ \sum_{C \in \mathcal{C}} C_k \cdot x_C &= N_k & \forall k \in \mathcal{K} \\ x_C &\in \mathbb{Z}_{\geq 0} & \forall C \in \mathcal{C} \end{aligned}$$

Here  $\mathcal{K}$  are the rounded sizes and  $N_k$  the number of jobs with rounded size  $k \in \mathcal{K}$ . The first constraint enforces that the correct number of machines is used, the next  $|\mathcal{K}|$  many enforce that for each size the correct number of jobs is scheduled.

It is notable that this ILP has only few constraints (a constant for a fixed choice of  $\epsilon$ ) and also the entries of the matrix are small. More precisely, they are at most  $1/\epsilon$ , since every size is at least  $\epsilon \cdot \tau$  and therefore no more than  $1/\epsilon$  jobs fit in one configuration. The ILP

can be solved with our algorithm. Note that  $\Delta \leq 1/\epsilon$ ,  $m = O(1/\epsilon \log(1/\epsilon))$ ,  $\|b\|_\infty \leq N$ , and  $n \leq (1/\epsilon)^{O(1/\epsilon \log(1/\epsilon))}$ . Including the rounding in time  $O(N + 1/\epsilon \log(1/\epsilon))$  the running time for the ILP is

$$\begin{aligned} O(m\Delta)^m \cdot \log(\Delta) \cdot \log(\Delta + \|b\|_\infty) + O(nm) + O(N + 1/\epsilon \log(1/\epsilon)) \\ \leq 2^{O(1/\epsilon \log^2(1/\epsilon))} \log(N) + O(N + 1/\epsilon \log(1/\epsilon)) \leq 2^{O(1/\epsilon \log^2(1/\epsilon))} + O(N). \end{aligned}$$

The trick in the bound above is to distinguish between the cases  $2^{O(1/\epsilon \log^2(1/\epsilon))} \leq \log(N)$  and  $2^{O(1/\epsilon \log^2(1/\epsilon))} > \log(N)$ . The same running time (except for a higher constant in the exponent) could be obtained with [8]. However, in order to avoid a multiplicative factor of  $N$ , one would have to solve the LP relaxation first and then use proximity. Our approach gives an easier, purely combinatorial algorithm. The crucial feature of our algorithm is the lower dependence on  $\|b\|_\infty$ .

---

## References

- 1 Amir Abboud, Karl Bringmann, Danny Hermelin, and Dvir Shabtay. SETH-Based Lower Bounds for Subset Sum and Bicriteria Path. *CoRR*, abs/1704.04546, 2017. [arXiv:1704.04546](#).
- 2 Kyriakos Axiotis and Christos Tzamos. Capacitated Dynamic Programming: Faster Knapsack and Graph Algorithms. *CoRR*, abs/1802.06440, 2018. [arXiv:1802.06440](#).
- 3 Arturs Backurs, Piotr Indyk, and Ludwig Schmidt. Better Approximations for Tree Sparsity in Nearly-Linear Time. In *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2017, Barcelona, Spain, Hotel Porta Fira, January 16-19*, pages 2215–2229, 2017. doi:10.1137/1.9781611974782.145.
- 4 David Bremner, Timothy M. Chan, Erik D. Demaine, Jeff Erickson, Ferran Hurtado, John Iacono, Stefan Langerman, Mihai Patrascu, and Perouz Taslakian. Necklaces, Convolutions, and X+Y. *Algorithmica*, 69(2):294–314, 2014. doi:10.1007/s00453-012-9734-3.
- 5 Karl Bringmann. A Near-Linear Pseudopolynomial Time Algorithm for Subset Sum. In *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2017, Barcelona, Spain, Hotel Porta Fira, January 16-19*, pages 1073–1084, 2017. doi:10.1137/1.9781611974782.69.
- 6 Timothy M. Chan and Moshe Lewenstein. Clustered Integer 3SUM via Additive Combinatorics. In *Proceedings of the Forty-Seventh Annual ACM on Symposium on Theory of Computing, STOC 2015, Portland, OR, USA, June 14-17, 2015*, pages 31–40, 2015. doi:10.1145/2746539.2746568.
- 7 Marek Cygan, Marcin Mucha, Karol Wegrzycki, and Michal Włodarczyk. On Problems Equivalent to (min, +)-Convolution. In *44th International Colloquium on Automata, Languages, and Programming, ICALP 2017, July 10-14, 2017, Warsaw, Poland*, pages 22:1–22:15, 2017. doi:10.4230/LIPIcs.ICALP.2017.22.
- 8 Friedrich Eisenbrand and Robert Weismantel. Proximity results and faster algorithms for Integer Programming using the Steinitz Lemma. In *Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2018, New Orleans, LA, USA, January 7-10, 2018*, pages 808–816, 2018. doi:10.1137/1.9781611975031.52.
- 9 Fedor V. Fomin, Fahad Panolan, M. S. Ramanujan, and Saket Saurabh. On the Optimality of Pseudo-polynomial Algorithms for Integer Programming. In *26th Annual European Symposium on Algorithms, ESA 2018, August 20-22, 2018, Helsinki, Finland*, pages 31:1–31:13, 2018. doi:10.4230/LIPIcs.ESA.2018.31.
- 10 Klaus Jansen, Kim-Manuel Klein, and José Verschae. Closing the Gap for Makespan Scheduling via Sparsification Techniques. In *43rd International Colloquium on Automata, Languages, and Programming, ICALP 2016, July 11-15, 2016, Rome, Italy*, pages 72:1–72:13, 2016. doi:10.4230/LIPIcs.ICALP.2016.72.

- 11 Hendrik W. Lenstra Jr. Integer Programming with a Fixed Number of Variables. *Math. Oper. Res.*, 8(4):538–548, 1983. doi:10.1287/moor.8.4.538.
- 12 Ravi Kannan. Minkowski’s Convex Body Theorem and Integer Programming. *Math. Oper. Res.*, 12(3):415–440, 1987. doi:10.1287/moor.12.3.415.
- 13 Hans Kellerer, Ulrich Pferschy, and David Pisinger. *Knapsack problems*. Springer, 2004.
- 14 Eduardo Sany Laber, Wilfredo Bardales Roncalla, and Ferdinando Cicalese. On lower bounds for the Maximum Consecutive Subsums Problem and the (min, +)-convolution. In *2014 IEEE International Symposium on Information Theory, Honolulu, HI, USA, June 29 - July 4, 2014*, pages 1807–1811, 2014. doi:10.1109/ISIT.2014.6875145.
- 15 Christos H. Papadimitriou. On the complexity of integer programming. *J. ACM*, 28(4):765–768, 1981. doi:10.1145/322276.322287.
- 16 Sergey Vasil’evich Sevast’janov. Approximate solution of some problems in scheduling theory. *Metody Diskret. Analiz*, 32:66–75, 1978. in Russian.
- 17 Ernst Steinitz. Bedingt konvergente Reihen und konvexe Systeme. *Journal für die reine und angewandte Mathematik*, 143:128–176, 1913.



# Empowering the Configuration-IP – New PTAS Results for Scheduling with Setups Times

**Klaus Jansen**<sup>1</sup>

Department of Computer Science, Kiel University, Kiel, Germany  
kj@informatik.uni-kiel.de

**Kim-Manuel Klein**

Department of Computer Science, Kiel University, Kiel, Germany  
kmk@informatik.uni-kiel.de

**Marten Maack**

Department of Computer Science, Kiel University, Kiel, Germany  
mmaa@informatik.uni-kiel.de

**Malin Rau**

Department of Computer Science, Kiel University, Kiel, Germany  
mra@informatik.uni-kiel.de

---

## Abstract

Integer linear programs of configurations, or configuration IPs, are a classical tool in the design of algorithms for scheduling and packing problems, where a set of items has to be placed in multiple target locations. Herein a configuration describes a possible placement on one of the target locations, and the IP is used to choose suitable configurations covering the items. We give an augmented IP formulation, which we call the module configuration IP. It can be described within the framework of  $n$ -fold integer programming and therefore be solved efficiently. As an application, we consider scheduling problems with setup times, in which a set of jobs has to be scheduled on a set of identical machines, with the objective of minimizing the makespan. For instance, we investigate the case that jobs can be split and scheduled on multiple machines. However, before a part of a job can be processed an uninterrupted setup depending on the job has to be paid. For both of the variants that jobs can be executed in parallel or not, we obtain an efficient polynomial time approximation scheme (EPTAS) of running time  $f(1/\varepsilon) \times \text{poly}(|I|)$  with a single exponential term in  $f$  for the first and a double exponential one for the second case. Previously, only constant factor approximations of  $5/3$  and  $4/3 + \varepsilon$  respectively were known. Furthermore, we present an EPTAS for a problem where classes of (non-splittable) jobs are given, and a setup has to be paid for each class of jobs being executed on one machine.

**2012 ACM Subject Classification** Theory of computation → Scheduling algorithms, Theory of computation → Discrete optimization

**Keywords and phrases** Parallel Machines, Setup Time, EPTAS,  $n$ -fold integer programming

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2019.44

**Related Version** The long version is hosted on arXiv [14], <https://arxiv.org/abs/1801.06460>.

**Acknowledgements** We thank Syamantak Das for helpful discussions on the problem.

---

<sup>1</sup> German Research Foundation (DFG) project JA 612/20-1





## 1 Introduction

In this paper, we present an augmented formulation of the classical integer linear program of configurations (configuration IP) and demonstrate its use in the design of efficient polynomial time approximation schemes for scheduling problems with setup times. Configuration IPs are widely used in the context of scheduling or packing problems, in which items have to be distributed to multiple target locations. The configurations describe possible placements on a single location, and the integer linear program (IP) is used to choose a proper selection covering all items. Two fundamental problems, for which configuration IPs have prominently been used, are bin packing and minimum makespan scheduling on identical parallel machines, or machine scheduling for short. For bin packing, the configuration IP was introduced as early as 1961 by Gilmore and Gomory [10], and the recent results for both problems typically use configuration IPs as a core technique, see, e.g., [11, 15]. In the present work, we consider scheduling problems and therefore introduce the configuration IP in more detail using the example of machine scheduling.

**Configuration IP for Machine Scheduling.** In the problem of machine scheduling, a set  $\mathcal{J}$  of  $n$  jobs is given together with processing times  $p_j$  for each job  $j$  and a number  $m$  of identical machines. The objective is to find a schedule  $\sigma : \mathcal{J} \rightarrow [m]$ , such that the makespan is minimized, that is, the latest finishing time of any job  $C_{\max}(\sigma) = \max_{i \in [m]} \sum_{j \in \sigma^{-1}(i)} p_j$ . For a given makespan bound, the configurations may be defined as multiplicity vectors indexed by the occurring processing times, where the overall length of the chosen processing times does not violate the bound. The configuration IP is then given by variables  $x_C$  for each configuration  $C$ ; constraints ensuring that there is a machine for each configuration, i.e.,  $\sum_C x_C = m$ ; and further constraints due to which the jobs are covered, i.e.,  $\sum_C C_p x_C = |\{j \in \mathcal{J} \mid p_j = p\}|$  for each processing time  $p$ . In combination with certain simplification techniques, this type of IP is often used in the design of *polynomial time approximation schemes* (PTAS). A PTAS is a procedure that, for any fixed accuracy parameter  $\varepsilon > 0$ , returns a solution with approximation guarantee  $(1 + \varepsilon)$  that is, a solution, whose objective value lies within a factor of  $(1 + \varepsilon)$  of the optimum. In the context of machine scheduling, the aforementioned simplification techniques can be used to guess the target makespan  $T$  of the given instance; to upper bound the cardinality of the set of processing times  $P$  by a constant (depending in  $1/\varepsilon$ ); and to lower bound the processing times in size, such that they are within a constant factor of the makespan  $T$  (see, e.g., [3, 15]). Hence, only a constant number of configurations is needed, yielding an integer program with a constant number of variables. Integer programs of that kind can be efficiently solved using the classical algorithm by Lenstra and Kannan [21, 17], yielding a PTAS for machine scheduling. Here, the error of  $(1 + \varepsilon)$  in the quality of the solution is due to the simplification steps, and the scheme has a running time of the form  $f(1/\varepsilon) \times \text{poly}(|I|)$ , where  $|I|$  denotes the input size, and  $f$  some computable function. A PTAS with this property is called *efficient* (EPTAS). Note that for a regular PTAS a running time of the form  $|I|^{f(1/\varepsilon)}$  is allowed. It is well known, that machine scheduling is strongly NP-hard, and therefore there is no optimal polynomial time algorithm, unless  $P=NP$ , and also a so-called *fully polynomial* PTAS (FPTAS) – which is an EPTAS with a polynomial function  $f$  – cannot be hoped for.

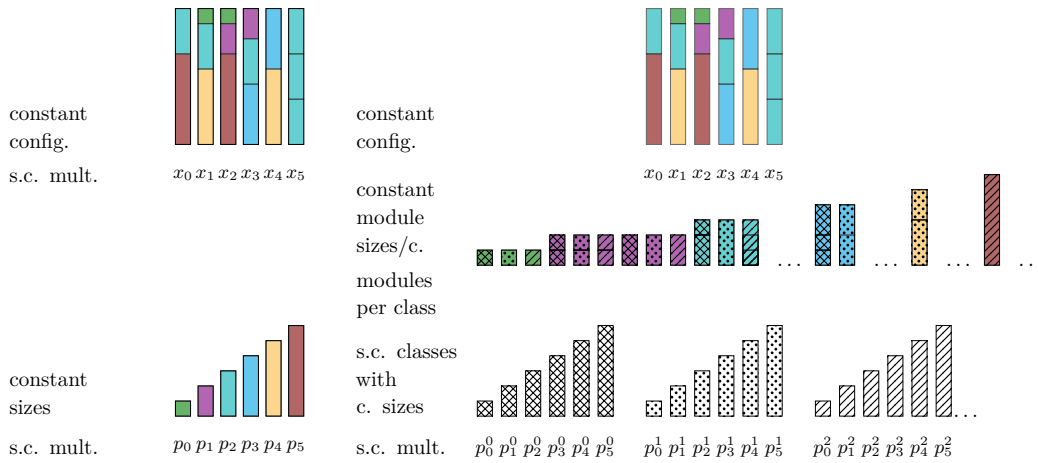
**Machine Scheduling with Classes.** The configuration IP is used in a wide variety of approximation schemes for machine scheduling problems [3, 15]. However, the approach often ceases to work for scheduling problems in which the jobs have to fulfill some additional

requirements, like, for instance, class dependencies. A problem emerging, in this case, is that the additional requirements have to be represented in the configurations, resulting in a super-constant number of variables in the IP. We elaborate on this using a concrete example: Consider the variant of machine scheduling in which the jobs are partitioned into  $K$  setup classes. For each job  $j$  a class  $k_j$  is given and for each class  $k$  a setup time  $s_k$  has to be paid on a machine, if a job belonging to that class is scheduled on it, i.e.,  $C_{\max}(\sigma) = \max_{i \in [m]} \left( \sum_{j \in \sigma^{-1}(i)} p_j + \sum_{k \in \{k_j \mid j \in \sigma^{-1}(i)\}} s_k \right)$ . With some effort, simplification steps similar to the ones for machine scheduling can be applied. In the course of this, the setup times as well can be bounded in number and guaranteed to be sufficiently big [16]. However, it is not hard to see that the configuration IP still cannot be trivially extended, while preserving its solvability. For instance, extending the configurations with multiplicities of setup times will not work, because then we have to make sure that a configuration is used for a fitting subset of classes, creating the need to encode class information into the configurations or introduce other class dependent variables.

**Module Configuration IP.** Our approach to deal with the class dependencies of the jobs is to cover the job classes with so-called modules and cover the modules in turn with configurations in an augmented IP called the module configuration IP (MCIP). In the setup class model, for instance, the modules may be defined as combinations of setup times and configurations of processing times, and the actual configurations as multiplicity vectors of module sizes. The number of both the modules and the configurations will typically be bounded by a constant. To cover the classes by modules each class is provided with its own set of modules, that is, there are variables for each pair of class and module. Since the number of classes is part of the input, the number of variables in the resulting MCIP is super-constant, and therefore the algorithm by Lenstra and Kannan [21, 17] is not the proper tool for the solving of the MCIP. However, the MCIP has a certain simple structure: The mentioned variables are partitioned into uniform classes each corresponding to the set of modules, and for each class, the modules have to do essentially the same – cover the jobs of the class. Utilizing these properties, we can formulate the MCIP in the framework of  $n$ -fold integer programs – a class of IPs whose variables and constraints fulfill certain uniformity requirements. In 2013 Hemmecke, Onn, and Romanchuk [12] showed that  $n$ -fold IPs can be efficiently solved, and very recently both Eisenbrand, Hunkenschroder and Klein [9] and independently Koucký, Levin and Onn [20] developed algorithms with greatly improved running times for the problem. For a detailed description of the MCIP, the reader is referred to Section 3. In Figure 1 the basic idea of the MCIP is visualized.

Using the MCIP, we are able to formulate an EPTAS for machine scheduling in the setup class model described above. Before, only a regular PTAS with running time  $nm^{\mathcal{O}(1/\varepsilon^5)}$  was known [16]. To the best of our knowledge, this is the first use of  $n$ -fold integer programming in the context of approximation algorithms.

**Results and Methodology.** To show the conceptual power of the MCIP, we utilize it for two more problems: The *splittable* and the *preemptive* setup model of machine scheduling. In both variants for each job  $j$ , a setup time  $s_j$  is given. Each job may be partitioned into multiple parts that can be assigned to different machines, but before any part of the job can be processed the setup time has to be paid. In the splittable model, job parts belonging to the same job can be processed in parallel, and therefore beside the partition of the jobs, it suffices to find an assignment of the job parts to machines. This is not the case for the preemptive model, in which additionally a starting time for each job part has to be found, and two



■ **Figure 1** On the left, there is a schematic representation of the configuration IP. There are constant different sizes each occurring a super-constant number of times. The sizes are directly mapped to configurations. On the right, there is a schematic representation of the MCIP. There is a super-constant number of classes, each containing a constant number of sizes which have super-constant multiplicities. The elements from the class are mapped to a constant number of different modules, which have a constant number of sizes. These module sizes are mapped to configurations.

parts of the same job may not be processed in parallel. In 1999 Schuurman and Woeginger [26] presented a polynomial time algorithm for the preemptive model with approximation guarantee  $4/3 + \epsilon$ , and for the splittable case a guarantee of  $5/3$  was achieved by Chen, Ye and Zhang [5]. These are the best known approximation guarantees for the problems at hand. We show that solutions arbitrarily close to the optimum can be found in polynomial time:

► **Theorem 1.** *There is an efficient PTAS with running time  $2^{f(1/\epsilon)} \text{poly}(|I|)$  for minimum makespan scheduling on identical parallel machines in the setup-class model, as well as in the preemptive and splittable setup models.*

More precisely, we get a running time of  $2^{\mathcal{O}(1/\epsilon^3 \log^4 1/\epsilon)} K^2 nm \log(Km)$  in the setup class model,  $2^{\mathcal{O}(1/\epsilon^2 \log^3 1/\epsilon)} n^2 \log^3(nm)$  in the splittable, and  $2^{2^{\mathcal{O}(1/\epsilon \log 1/\epsilon)}} n^2 m \log m \log(nm)$  in the preemptive model (remember that  $n$ ,  $m$ , and  $K$  denote the number of jobs, machines, and setup classes). Note that all three problems are strongly NP-hard, due to trivial reductions from machine scheduling, and our results are therefore in some sense best possible.

Summing up, the main achievement of this work is the development of the module configuration IP and its application in the development of approximation schemes. Up to now, EPTAS or even PTAS results seemed out of reach for the considered problems, and for the preemptive model we provide the first improvement in 20 years. The simplification techniques developed for the splittable and preemptive model in order to employ the MCIP are original and in the latter case quite elaborate, and therefore interesting by themselves. Furthermore, we expect the MCIP to be applicable to other packing and scheduling problems as well, in particular for variants of machine scheduling and bin packing with additional class depended constraints. On a more conceptual level, we gave a first demonstration of the potential of  $n$ -fold integer programming in the theory of approximation algorithms, and hope to inspire further studies in this direction.

We conclude this paragraph with a more detailed overview of our results and their presentation. For all three EPTAS results we employ the classical dual approximation framework by Hochbaum and Shmoys [13] to get a guess of the makespan  $T$ . This approach is

introduced in Section 2 together with  $n$ -fold IPs and formal definitions of the problems. In the following section, we develop the module configuration IP, in its basic form and argue that it is indeed an  $n$ -fold IP. The EPTAS results described in the last section follow the same basic approach described above for machine scheduling: We find a schedule for a simplified instance via the MCIP and transform it into a schedule for the original one. The simplification steps typically include rounding of the processing and setup times using standard techniques, as well as, the removal of certain jobs, which later can be reinserted via carefully selected greedy procedures. For the splittable and preemptive model, we additionally have to prove that schedules with a certain simple structure exist, and in the preemptive model, the MCIP has to be extended. Missing proofs and details can be found in the long version of the paper [14].

**Related work.** For an overview on  $n$ -fold IPs and their applications, we refer to the book by Onn [24]. There have been recent applications of  $n$ -fold integer programming to scheduling problems in the context of parameterized algorithms: Knop and Koutecký [18] showed, among other things, that the problem of makespan minimization on unrelated parallel machines, where the processing times are dependent on both jobs and machines, is fixed-parameter tractable with respect to the maximum processing time and the number of distinct machine types. This was generalized to the parameters maximum processing time and rank of the processing time matrix by Chen et al. [6]. Furthermore, Knop, Koutecký and Mnich [19] provided an improved algorithm for a special type of  $n$ -fold IPs yielding improved running times for several applications of  $n$ -fold IPs including results for scheduling problems.

There is extensive literature concerning scheduling problems with setup times. We highlight a few closely related results and otherwise refer to the surveys [1, 2]. In the following, we use the term  $\alpha$ -approximation as an abbreviation for polynomial time algorithms with approximation guarantee  $\alpha$ . The setup class model was first considered by Mäcker et al. [22] in the special case that all classes have the same setup time. They designed a 2-approximation and additionally a  $3/2 + \varepsilon$ -approximation for the case that the overall length of the jobs from each class is bounded. Jansen and Land [16] presented a simple 3-approximation with linear running time, a  $2 + \varepsilon$ -approximation, and the aforementioned PTAS for the general setup class model. As indicated before, Chen et al. [5] developed a  $5/3$ -approximation for the splittable model. A generalization of this, in which both setup and processing times are job and machine dependent, has been considered by Correa et al. [7]. They achieve a  $(1 + \phi)$ -approximation, where  $\phi$  denotes the golden ratio, using a newly designed linear programming formulation. Moreover, there are recent results concerning machine scheduling in the splittable model considering the sum of the (weighted) completion times as the objective function, e.g. [25, 8]. For the preemptive model, a PTAS for the special case that all jobs have the same setup time has been developed by Schuurman and Woeginger [26]. The mentioned  $(4/3 + \varepsilon)$ -approximation for the general case [26] follows the same approach. Furthermore, a combination of the setup class and the preemptive model has been considered, in which the jobs are scheduled preemptively, but the setup times are class dependent. Monma and Potts [23] presented, among other things, a  $(2 - 1/(\lfloor m/2 \rfloor + 1))$ -approximation for this model, and later Chen [4] achieved improvements for some special cases.

## 2 Preliminaries

In the following, we establish some concepts and notations, formally define the considered problems, and outline the dual approximation approach by Hochbaum and Shmoys [13], as well as  $n$ -fold integer programs.

For any integer  $n$ , we denote the set  $\{1, \dots, n\}$  by  $[n]$ ; we write  $\log(\cdot)$  for the logarithm with basis 2; and we will usually assume that some instance  $I$  of the problem considered in the respective context is given together with an accuracy parameter  $\varepsilon \in (0, 1)$  such that  $1/\varepsilon$  is an integer. Furthermore for any two sets  $X, Y$  we write  $Y^X$  for the set of functions  $f : X \rightarrow Y$ . If  $X$  is finite, we say that  $Y$  is indexed by  $X$  and sometimes denote the function value of  $f$  for the argument  $x \in X$  by  $f_x$ .

**Problems.** For all three of the considered models, a set  $\mathcal{J}$  of  $n$  jobs with processing times  $p_j \in \mathbb{Q}_{>0}$  for each job  $j \in \mathcal{J}$  and a number of machines  $m$  is given. In the preemptive and the splittable model, the input additionally includes a setup time  $s_j \in \mathbb{Q}_{>0}$  for each job  $j \in \mathcal{J}$ ; while in the setup class model, it includes a number  $K$  of setup classes, a setup class  $k_j \in [K]$  for each job  $j \in \mathcal{J}$ , as well as setup times  $s_k \in \mathbb{Q}_{>0}$  for each  $k \in [K]$ .

We take a closer look at the definition of a schedule in the preemptive model. The jobs may be split. Therefore, partition sizes  $\kappa : \mathcal{J} \rightarrow \mathbb{Z}_{>0}$ , together with processing time fractions  $\lambda_j : [\kappa(j)] \rightarrow (0, 1]$ , such that  $\sum_{k \in [\kappa(j)]} \lambda_j(k) = 1$ , have to be found, meaning that job  $j$  is split into  $\kappa(j)$  many parts and the  $k$ -th part for  $k \in [\kappa(j)]$  has processing time  $\lambda_j(k)p_j$ . This given, we define  $\mathcal{J}' = \{(j, k) \mid j \in \mathcal{J}, k \in [\kappa(j)]\}$  to be the set of job parts. Now, an assignment  $\sigma : \mathcal{J}' \rightarrow [m]$  along with starting times  $\xi : \mathcal{J}' \rightarrow \mathbb{Q}_{>0}$  has to be determined, such that any two job parts assigned to the same machine or belonging to the same job do not overlap. More precisely, we have to assure that for each two job parts  $(j, k), (j', k') \in \mathcal{J}'$  with  $\sigma(j, k) = \sigma(j', k')$  or  $j = j'$ , we have  $\xi(j, k) + s_j + \lambda_j(k)p_j \leq \xi(j')$  or  $\xi(j', k') + s_{j'} + \lambda_{j'}(k')p_{j'} \leq \xi(j)$ . A schedule is given by  $(\kappa, \lambda, \sigma, \xi)$  and the makespan can be defined as  $C_{\max} = \max_{(j,k) \in \mathcal{J}'} (\xi(j, k) + s_j + \lambda_j(k)p_j)$ . Note that the variant of the problem in which overlap between a job part and setup of the same job is allowed is equivalent to the one presented above. This was pointed out by Schuurmann and Woeginger [26] and can be seen with a simple swapping argument.

In the splittable model, it is not necessary to determine starting times for the job parts, because, given the assignment  $\sigma$ , the job parts assigned to each machine can be scheduled as soon as possible in arbitrary order without gaps. Hence, in this case, the output is of the form  $(\kappa, \lambda, \sigma)$  and the makespan can be defined as  $C_{\max} = \max_{i \in [m]} \sum_{(j,k) \in \sigma^{-1}(i)} s_j + \lambda_j(k)p_j$ .

Lastly, in the setup class model the jobs are not split and given an assignment, the jobs assigned to each machine can be scheduled in batches comprised of the jobs of the same class assigned to the machine without overlaps and gaps. The output is therefore just an assignment  $\sigma : \mathcal{J} \rightarrow [m]$  and the makespan is given by  $C_{\max} = \max_{i \in [m]} \sum_{j \in \sigma^{-1}(i)} p_j + \sum_{k \in \{k_j \mid j \in \sigma^{-1}(i)\}} s_k$ .

Note that in the preemptive and the setup class model, we can assume that the number of machines is bounded by the number of jobs: If there are more machines than jobs, placing each job on a private machine yields an optimal schedule in both models and the remaining machines can be ignored. This, however, is not the case in the splittable model.

**Dual Approximation.** All of the presented algorithms follow the dual approximation framework introduced by Hochbaum and Shmoys [13]: Instead of solving the minimization version of a problem directly, it suffices to find a procedure that for a given bound  $T$  on the objective value either correctly reports that there is no solution with value  $T$  or returns a solution with value at most  $(1 + a\varepsilon)T$  for some constant  $a$ . If we have some initial upper bound  $B$  for the optimal makespan  $\text{OPT}$  with  $B \leq b\text{OPT}$  for some  $b$ , we can define a PTAS by trying different values  $T$  from the interval  $[B/b, B]$  in a binary search fashion, and find a value  $T^* \leq (1 + \mathcal{O}(\varepsilon))\text{OPT}$  after  $\mathcal{O}(\log b/\varepsilon)$  iterations. Note that for all of the considered problems

there are known simple approximation algorithms. Hence, we always assume that a target makespan  $T$  is given. Furthermore, we assume that the setup times and in the preemptive and setup class cases also the processing times are bounded by  $T$ , because otherwise we can reject  $T$  immediately.

**$n$ -fold Integer Programs.** We briefly define  $n$ -fold integer programs (IP) following the notation of [12] and [18] and state the main algorithmic result needed in the following. Let  $n, r, s, t \in \mathbb{Z}_{>0}$  be integers and  $A$  be an integer  $((r + ns) \times nt)$ -matrix of the following form:

$$A = \begin{pmatrix} A_1 & A_1 & \cdots & A_1 \\ A_2 & 0 & \cdots & 0 \\ 0 & A_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & A_2 \end{pmatrix}$$

The matrix  $A$  is the so-called  $n$ -fold product of the bimatrix  $\begin{pmatrix} A_1 \\ A_2 \end{pmatrix}$ , with  $A_1$  an  $r \times t$  and  $A_2$  an  $s \times t$  matrix. Furthermore, let  $w, \ell, u \in \mathbb{Z}^{nt}$  and  $b \in \mathbb{Z}^{r+ns}$ . Then the  $n$ -fold integer programming problem is given by:

$$\min\{wx \mid Ax = b, \ell \leq x \leq u, x \in \mathbb{Z}^{nt}\}$$

The variables  $x$  can naturally be partitioned into *bricks*  $x^{(a)}$  of dimension  $t$  for each  $q \in [n]$ , such that  $x = (x^{(1)}, \dots, x^{(n)})$ . Furthermore, we denote the constraints corresponding to  $A_1$  as *globally uniform* and the ones corresponding to  $A_2$  as *locally uniform*. Hence,  $r$  is the number of globally and  $s$  the number of locally uniform constraints (ignoring their  $n$ -fold duplication);  $t$  the *brick size* and  $n$  the *brick number*. We set  $\Delta$  to be the maximum absolute value occurring in  $A$ . Up to recently the best known algorithm for solving  $n$ -fold IPs was due to Hemmecke, Onn and Romanchuk [12]:

► **Theorem 2.** *Let  $\varphi$  be the encoding length of  $w, b, \ell, u$  and  $\Delta$ . The  $n$ -fold integer programming problem can be solved in time  $\mathcal{O}(\Delta^{3t(rs+st+r+s)}n^3\varphi)$ , when  $r, s$  and  $t$  are fixed.*

However, in 2018 both Eisenbrand, Hunkenschröder and Klein [20] and independently Koutecký, Levin and Onn [20] developed algorithms with improved and very similar running times. We state a variant due to Eisenbrand et al. that is adapted to our needs:

► **Theorem 3.** *Let  $\varphi$  be the encoding length of the largest number occurring in the input, and  $\Phi = \max_i(u_i - \ell_i)$ . The  $n$ -fold integer programming problem can be solved in time  $(rs\Delta)^{\mathcal{O}(r^2s+rs^2)}t^2n^2\varphi \log(\Phi) \log(nt\Phi)$ .*

### 3 Module Configuration IP

In this section, we state the configuration IP for machine scheduling and introduce a basic version of the module configuration IP (MCIP) that is already sufficiently general to work for both the splittable and setup class model. Before that, however, we formally introduce the concept of *configurations*. Given a set of objects  $A$ , a configuration  $C$  of these objects is a vector of multiplicities indexed by the objects, i.e.,  $C \in \mathbb{Z}_{\geq 0}^A$ . For given sizes  $\Lambda(a)$  of the objects  $a \in A$ , the size  $\Lambda(C)$  of a configuration  $C$  is defined as  $\sum_{a \in A} C_a \Lambda(a)$ . Moreover, for a given bound  $B$ , we define  $\mathcal{C}_A(B)$  to be the set of configurations of  $A$  that are bounded in size by  $B$ , that is,  $\mathcal{C}_A(B) = \{C \in \mathbb{Z}_{\geq 0}^A \mid \Lambda(C) \leq B\}$ .

**Configuration IP.** We give a recollection of the configuration IP for scheduling on identical parallel machines. Let  $P$  be the set of processing times for some instance  $I$  with multiplicities  $n_p$  for each  $p \in P$ , meaning,  $I$  includes exactly  $n_p$  jobs with processing time  $p$  ( $\Lambda(p) = p$  in this context). Furthermore, let  $T$  be a guess of the optimal makespan. The configuration IP for  $I$  and  $T$  is given by variables  $x_C \geq 0$  for each  $C \in \mathcal{C}_P(T)$  and the following constraints:

$$\sum_{C \in \mathcal{C}_P(T)} x_C = m \quad (1)$$

$$\sum_{C \in \mathcal{C}_P(T)} C_p x_C = n_p \quad \forall p \in P \quad (2)$$

Due to constraint (1), exactly one configuration is chosen for each machine, while (2) ensures that the correct number of jobs or job sizes is covered.

**Module Configuration IP.** Let  $\mathcal{B}$  be a set of basic objects (e.g. jobs or setup classes) and let there be  $D$  integer values  $B_1, \dots, B_D$  for each basic object  $B \in \mathcal{B}$  (e.g. processing time or numbers of different kinds of jobs). Our approach is to cover the basic objects with so-called *modules*  $\mathcal{M}$  and in turn cover the modules with configurations. Depending on the context, modules correspond to batches of jobs or job piece sizes together with a setup time and can also encompass additional information like a starting time. Corresponding to the basic objects, each module  $M \in \mathcal{M}$  also has  $D$  integer values  $M_1, \dots, M_D$ , as well as a size  $\Lambda(M)$  and a set of eligible basic objects  $\mathcal{B}(M)$ . The latter is needed because not all modules are compatible with all basic objects, e.g., because they do not have the right setup times. Let  $H$  be the set of distinct module sizes, i.e.,  $H = \{\Lambda(M) \mid M \in \mathcal{M}\}$ , and for each module size  $h \in H$  let  $\mathcal{M}(h)$  be the set of modules with size  $h$ . We consider the set  $\mathcal{C}$  of configurations of module sizes which are bounded in size by a guess of the makespan  $T$ , i.e.,  $\mathcal{C} = \mathcal{C}_H(T)$ . In the preemptive case configurations need to additionally encompass information about starting times of modules, and therefore the definition of configurations will be slightly more complicated in that case.

Since we want to chose configurations for each machine, we have variables  $x_C$  for each  $C \in \mathcal{C}$  and constraints corresponding to (1). Furthermore, we chose modules with variables  $y_M$  for each  $M \in \mathcal{M}$  and because we want to cover the chosen modules with configurations, we have some analogue of constraint (2), say  $\sum_{C \in \mathcal{C}(T)} C_h x_C = \sum_{M \in \mathcal{M}(h)} y_M$  for each module size  $h \in H$ . It turns out however, that to properly cover the basic objects with modules, we need the variables  $y_M$  for each basic object, and this is where  $n$ -fold IPs come into play: The variables stated so far form a brick of the variables of the  $n$ -fold IP and there is one brick for each basic object, that is, we have, for each  $B \in \mathcal{B}$ , variables  $x_C^{(B)}$  for each  $C \in \mathcal{C}$ , and  $y_M^{(B)}$  for each  $M \in \mathcal{M}$ . Using the upper bounds of the  $n$ -fold model, variables  $y_M^{(B)}$  are set to zero, if  $B$  is not eligible for  $M$ ; and we set the lower bounds of all variables to zero. Sensible upper bounds for the remaining variables, will be typically clear from context. Besides that, the module configuration integer program MCIP (for  $\mathcal{B}$ ,  $\mathcal{M}$  and  $\mathcal{C}$ ) is given by the following constraints.

$$\sum_{B \in \mathcal{B}} \sum_{C \in \mathcal{C}} x_C^{(B)} = m \quad (3)$$

$$\sum_{B \in \mathcal{B}} \sum_{C \in \mathcal{C}(T)} C_h x_C^{(B)} = \sum_{B \in \mathcal{B}} \sum_{M \in \mathcal{M}(h)} y_M^{(B)} \quad \forall h \in H \quad (4)$$

$$\sum_{M \in \mathcal{M}} M_d y_M^{(B)} = B_d \quad \forall B \in \mathcal{B}, d \in [D] \quad (5)$$



It is easy to see that the constraints (3) and (4) are globally uniform. They are the mentioned adaptations of (1) and (2). The constraint (5), on the other hand, is locally uniform and ensures that the basic objects are covered.

Note that, while the duplication of the configuration variables does not carry meaning, it also does not upset the model: Consider the modified MCIP that is given by not duplicating the configuration variables. A solution  $(\tilde{x}, \tilde{y})$  for this IP gives a solution  $(x, y)$  for the MCIP by fixing some basic object  $B^*$ , setting  $x_C^{(B^*)} = \tilde{x}_C$  for each configuration  $C$ , setting the remaining configuration variables to 0, and copying the remaining variables. Given a solution  $(x, y)$  for the MCIP, on the other hand, gives a solution for the modified version  $(\tilde{x}, \tilde{y})$  by setting  $\tilde{x}_C = \sum_{B \in \mathcal{B}} x_C^B$  for each configuration  $C$ . Summarizing we get:

► **Observation 1.** *The MCIP is an  $n$ -fold IP with brick-size  $t = |\mathcal{M}| + |\mathcal{C}|$ , brick number  $n = |\mathcal{B}|$ ,  $r = |H| + 1$  globally uniform and  $s = D$  locally uniform constraints.*

Moreover, in all the considered applications we will minimize the overall size of the configurations, i.e.,  $\sum_{B \in \mathcal{B}} \sum_{C \in \mathcal{C}} \Lambda(C) x_C^{(B)}$ . This will be required, because in the simplification steps of our algorithms some jobs are removed and have to be reinserted later, and we therefore have to make sure that no space is wasted.

## 4 EPTAS results

In this section, we present approximation schemes for each of the three considered problems. Each of the results follows the same approach: The instance is carefully simplified, a schedule for the simplified instance is found using the MCIP, and this schedule is transformed into a schedule for the original instance. The presentation of the result is also similar for each problem: We first discuss how the instance can be sensibly simplified, and how a schedule for the simplified instance can be transformed into a schedule for the original one. Next, we discuss how a schedule for the simplified instance can be found using the MCIP, and lastly, we summarize and analyze the taken steps.

For the sake of clarity, we have given rather formal definitions for the problems at hand in Section 2. In the following, however, we will use the terms in a more intuitive fashion for the most part, and we will, for instance, often take a geometric rather than a temporal view on schedules and talk about the *length* or the *space* taken up by jobs and setups on machines rather than time. In particular, given a schedule for an instance of any one of the three problems together with an upper bound for the makespan  $T$ , the *free space* with respect to  $T$  on a machine is defined as the summed up lengths of time intervals between 0 and  $T$  in which the machine is idle. The free space (with respect to  $T$ ) is the summed up free space of all the machines. For bounds  $T$  and  $L$  for the makespan and the free space, we say that a schedule is a  $(T, L)$ -schedule if its makespan is at most  $T$  and the free space with respect to  $T$  is at least  $L$ .

When transforming the instance we will increase or decrease processing and setup times and fill in or remove extra jobs. Consider a  $(T', L')$ -schedule, where  $T'$  and  $L'$  denote some arbitrary makespan or free space bounds. If we fill in extra jobs or increase processing or setup times, but can bound the increase on each machine by some bound  $b$ , we end up with a  $(T' + b, L')$ -schedule for the transformed instance. In particular we have the same bound for the free space, because we properly increased the makespan bound. If, on the other hand, jobs are removed or setup times decreased, we obviously still have a  $(T', L')$ -schedule for the transformed instance. This will be used frequently in the following.

### 4.1 Setup Class Model

In the setup class model, simplification steps similar to the ones developed by Jansen and Land [16] can be used. We give a brief overview:

The set of big setup jobs  $\mathcal{J}^{\text{bst}}$  is given by the jobs belonging to classes with setup times at least  $\varepsilon^3 T$  and the small setup jobs  $\mathcal{J}^{\text{sst}}$  are all the others. Furthermore, we call a job *tiny* or *small*, if its processing time is smaller than  $\varepsilon^4 T$  or  $\varepsilon T$  respectively, and *big* or *large* otherwise. For any given set of jobs  $J$ , we denote the subset of tiny jobs from  $J$  with  $J_{\text{tiny}}$  and the small, big and large jobs analogously. We simplify the instance in four steps, aiming for an instance that exclusively includes big jobs with big setup times and additionally only a constant number of distinct processing and setup times: We remove the small jobs with small setup times  $\mathcal{J}_{\text{small}}^{\text{sst}}$ ; increase the setup times of the remaining classes with small setup times to  $\varepsilon^3 T$ ; replace the tiny jobs from  $\mathcal{J}_{\text{tiny}}^{\text{bst}}$  with placeholders of size  $\varepsilon^4 T$ ; and round up the resulting processing and setup times in a geometric and a subsequent arithmetic rounding step. The above steps yield certain makespan bounds  $\bar{T} \leq \check{T} = (1 + \mathcal{O}(\varepsilon))T$ , and for the resulting instance  $I'$ , the sets  $P$  and  $S$  of distinct processing and setup times have bounded cardinality, that is,  $|P|, |S| \in \mathcal{O}(1/\varepsilon \log 1/\varepsilon)$ , and each processing and setup time is a multiple of  $\varepsilon^5 T$ . An obvious lower bound on the space taken up by the jobs from  $\mathcal{J}_{\text{small}}^{\text{sst}}$  in any schedule is given by  $L = \sum_{j \in \mathcal{J}_{\text{small}}^{\text{sst}}} p_j + \sum_{k \in Q} s_k$ .

► **Theorem 4.** *If there is a schedule for  $I$  with makespan  $T$ , there is a  $(\bar{T}, L)$ -schedule for  $I'$ ; and if there is a  $(\bar{T}, L)$ -schedule for  $I'$ , there is a schedule for  $I$  with makespan  $\check{T}$ . ◀*

**Utilization of the MCIP.** At this point, we can employ the module configuration IP. The basic objects in this context are the setup classes, i.e.,  $\mathcal{B} = [K']$ , and the different values are the numbers of jobs with a certain processing time, i.e.,  $D = |P|$ . We set  $n_{k,p}$  to be the number of jobs from setup class  $k \in [K']$  with processing time  $p \in P$ . The modules correspond to batches of jobs together with a setup time. Batches of jobs can be modeled as configurations of processing times, that is, multiplicity vectors indexed by the processing times. Hence, we define the set of modules  $\mathcal{M}$  to be the set of pairs of configurations of processing times and setup times with a summed up size bounded by  $\bar{T}$ , i.e.,  $\mathcal{M} = \{(C, s) \mid C \in \mathcal{C}_P(\bar{T}), s \in S, s + \Lambda(C) \leq \bar{T}\}$ , and write  $M_p = C_p$  and  $s_M = s$  for each module  $M = (C, s) \in \mathcal{M}$ . The values of a module  $M$  are given by the numbers  $M_p$  and its size  $\Lambda(M)$  by  $s_M + \sum_{p \in P} M_p p$ . Remember that the configurations  $\mathcal{C}$  are the configurations of module sizes  $H$  that are bounded in size by  $\bar{T}$ , i.e.,  $\mathcal{C} = \mathcal{C}_H(\bar{T})$ . A setup class is eligible for a module, if the setup times fit, i.e.,  $\mathcal{B}_M = \{k \in [K'] \mid s_k = s_M\}$ . Lastly, we establish  $\varepsilon^5 T = 1$  by scaling.

For the sake of readability, we state the resulting constraints of the MCIP with adapted notation and without duplication of the configuration variables:

$$\sum_{C \in \mathcal{C}} x_C = m \tag{6}$$

$$\sum_{C \in \mathcal{C}} C_h x_C = \sum_{k \in [K']} \sum_{M \in \mathcal{M}(h)} y_M^{(k)} \quad \forall h \in H \tag{7}$$

$$\sum_{M \in \mathcal{M}} M_p y_M^{(k)} = n_{k,p} \quad \forall k \in [K'], p \in P \tag{8}$$

Note that the coefficients are all integral and this includes those of the objective function, i.e.,  $\sum_C \Lambda(C) x_C$ , because of the scaling step.

► **Lemma 5.** *With the above definitions, there is a  $(\bar{T}, L)$ -schedule for  $I'$ , iff the MCIP has a solution with objective value at most  $m\bar{T} - L$ .*

**Proof.** Let there be a  $(\bar{T}, L)$ -schedule for  $I'$ . Then the schedule on a given machine corresponds to a distinct configuration  $C$  that can be determined by counting for each possible group size  $a$  the batches of jobs from the same class whose length together with the setup time adds up to an overall length of  $a$ . Note that the length of this configuration is equal to the used up space on that machine. We fix an arbitrary setup class  $k$  and set the variables  $x_C^{(k)}$  accordingly (and  $x_C^{(k')} = 0$  for  $k' \neq k$  and  $C \in \mathcal{C}$ ). By this setting, we get an objective value of at most  $m\bar{T} - L$  because there was  $L$  free space in the schedule. For each class  $k$  and module  $M$ , we count the number of machines on which there are exactly  $M_p$  jobs with processing time  $p$  from class  $k$  for each  $p \in P$ , and set  $y_M^{(k)}$  accordingly. It is easy to see that the constraints are satisfied by these definitions.

Given a solution  $(x, y)$  of the MCIP, we define a corresponding schedule: Because of (6), we can match the machines to configurations such that each machine is matched to exactly one configuration. If machine  $i$  is matched to  $C$ , we create  $C_G$  slots of length  $\Lambda(G)$  on  $i$  for each group  $G$ . Next, we divide the setup classes into batches. For each class  $k$  and module  $M$ , we create  $y_M^{(k)}$  batches of jobs from class  $k$  with  $M_p$  jobs with processing time  $p$  for each  $p \in P$  and place the batch together with the corresponding setup time into a fitting slot on some machine. Because of (8) and (7) all jobs can be placed by this process. Note that the used space equals the overall size of the configurations and we therefore have free space of at least  $L$ . ◀

**Result.** To analyze the running time of the resulting procedure, we mainly have to bound the parameters of the MCIP. We have  $|\mathcal{B}| = K' \leq K$  and  $D = |P|$  by definition, and  $|\mathcal{M}| = \mathcal{O}(|S|(1/\varepsilon^3)^{|P|}) = 2^{\mathcal{O}(1/\varepsilon \log^2 1/\varepsilon)}$ , because  $|S|, |P| \in \mathcal{O}(1/\varepsilon \log 1/\varepsilon)$ . This is true, due to the last rounding step, which also implies  $|H| \in \mathcal{O}(1/\varepsilon^5)$ , yielding  $|\mathcal{C}| = |H|^{\mathcal{O}(1/\varepsilon^3)} = 2^{\mathcal{O}(1/\varepsilon^3 \log 1/\varepsilon)}$ . According to Observation 1, this yields a brick size of  $t = 2^{\mathcal{O}(1/\varepsilon^3 \log 1/\varepsilon)}$ , a brick number of  $K$ ,  $\mathcal{O}(1/\varepsilon^5)$  globally, and  $\mathcal{O}(1/\varepsilon \log 1/\varepsilon)$  locally uniform constraints for the MCIP. We have  $\Delta = \mathcal{O}(1/\varepsilon^5)$ , because all occurring values in the processing time matrix are bounded in  $\bar{T}$ , and we have  $\bar{T} = \mathcal{O}(1/\varepsilon^5)$ , due to the scaling. Furthermore, the values of the objective function, the right hand side, and the upper and lower bounds on the variables are bounded by  $\mathcal{O}(n/\varepsilon^5)$ , yielding a bound of  $\mathcal{O}(\log n/\varepsilon^5)$  for the encoding length of the biggest number in the input  $\varphi$ . Lastly, all variables can be bounded by 0 from below and  $\mathcal{O}(m/\varepsilon^3)$  from above, yielding  $\Phi = \mathcal{O}(m/\varepsilon^3)$ .

By Theorem 3 and some arithmetic, the MCIP can be solved in time:

$$(rs\Delta)^{\mathcal{O}(r^2s+rs^2)} t^2 n^2 \varphi \log(\Phi) \log(nt\Phi) = 2^{\mathcal{O}(1/\varepsilon^{11} \log^2 1/\varepsilon)} K^2 \log n \log m \log Km$$

When building the actual schedule, we iterate through the jobs and machines, yielding an additional term that is linear in  $n$  and  $m$ . To get the running time stated below Theorem 1, additional rounding steps for the module sizes and slightly altered modules are needed. For details we refer to the long version of the paper.

## 4.2 Splittable Model

The approximation scheme for the splittable model presented in this section is probably the easiest one discussed in this work. There is, however, one problem concerning this procedure: Its running time is polynomial in the number of machines, which might be exponential in

the input size. In the long version, we show how this problem can be overcome and further improve the running time.

For the simplification of the instance, similar ideas to the setup class case can be employed. We give a brief overview:

In this context, the set of big setup jobs  $\mathcal{J}^{\text{bst}}$  is given by the jobs with setup times at least  $\varepsilon T$  and the small setup jobs  $\mathcal{J}^{\text{sst}}$  are all the others. Let  $L = \sum_{j \in \mathcal{J}^{\text{sst}}} (s_j + p_j)$ . Because every job has to be scheduled and every setup has to be paid at least once,  $L$  is a lower bound on the summed up space due to small jobs in any schedule. The instance is simplified in two steps: First, all the small setup jobs are removed, and next the remaining processing and setup times are rounded up to multiples of  $\varepsilon^2 T$ . These steps yield certain makespan bounds  $\bar{T} \leq \tilde{T} = (1 + \mathcal{O}(\varepsilon))T$ , and for the resulting instance  $I'$ , the sets  $P$  and  $S$  of distinct processing and setup times have bounded cardinality, that is,  $|P|, |S| \in \mathcal{O}(1/\varepsilon^2)$ .

► **Theorem 6.** *If there is a schedule with makespan  $T$  for  $I$ , there is also a  $(\bar{T}, L)$ -schedule for  $I'$  in which the length of each job part is a multiple of  $\varepsilon^2 T$ ; and if there is a  $(\bar{T}, L)$ -schedule for  $I'$ , there is also a schedule for  $I$  with makespan at most  $\tilde{T}$ .*

We will use the MCIP to find a  $(\bar{T}, L)$ -schedule for  $I'$  in which the length of each job part is a multiple of  $\varepsilon^2 T$ .

**Utilization of the MCIP.** The basic objects in this context are the (big setup) jobs, i.e.,  $\mathcal{B} = \mathcal{J}^{\text{bst}}$ , and they have only one value ( $D = 1$ ), namely, their processing time. Moreover, the modules are defined as the set of pairs of job piece sizes and setup times, i.e.,  $\mathcal{M} = \{(q, s) \mid s, q \in S \times P\}$ , and we write  $s_M = s$  and  $q_M = q$  for each module  $M = (q, s) \in \mathcal{M}$ . Corresponding to the value of the basic objects the value of a module  $M$  is  $q_M$ , and its size  $\Lambda(M)$  is given by  $q_M + s_M$ . A job is eligible for a module, if the setup times fit, i.e.,  $\mathcal{B}_M = \{j \in \mathcal{J}' \mid s_j = s_M\}$ . In order to ensure integral values, we establish  $\varepsilon^2 T = 1$  via scaling. The set of configurations  $\mathcal{C}$  is comprised of all configurations of module sizes  $H$  that are bounded in size by  $\bar{T}$ , i.e.,  $\mathcal{C} = \mathcal{C}_{\mathcal{M}}(\bar{T})$ . We state the constraints of the MCIP for the above definitions with adapted notation and without duplication of the configuration variables:

$$\sum_{C \in \mathcal{C}} x_C = m \tag{9}$$

$$\sum_{C \in \mathcal{C}} C_h x_C = \sum_{j \in \mathcal{J}'} \sum_{M \in \mathcal{M}(h)} y_M^{(j)} \quad \forall h \in H \tag{10}$$

$$\sum_{M \in \mathcal{M}} q_M y_M^{(j)} = p_j \quad \forall j \in \mathcal{J}' \tag{11}$$

Note that we additionally minimize the summed up size of the configurations, via the objective function  $\sum_C \Lambda(C) x_C$ .

► **Lemma 7.** *With the above definitions, there is a  $(\bar{T}, L)$ -schedule for  $I'$  in which the length of each job piece is a multiple of  $\varepsilon^2 T$ , iff MCIP has a solution with objective value at most  $m\bar{T} - L$ .*

**Proof.** Given such a schedule for  $I'$ , the schedule on each machine corresponds to exactly one configuration  $G$  that can be derived by counting the job pieces and setup times with the same summed up length  $a$  and setting  $C_G$  accordingly, where  $G$  is the group of modules with length  $a$ . The size of the configuration  $C$  is equal to the used space on the respective machine. Therefore, we can fix some arbitrary job  $j$  and set the variables  $x_C^{(j)}$  to the number of machines whose schedule corresponds to  $C$  (and  $x_C^{(j')} = 0$  for  $j' \neq j$  and  $C \in \mathcal{C}$ ). Since

there is at least a free space of  $L$  for the schedule, the objective value is bounded by  $m\bar{T} - L$ . Furthermore, for each job  $j$  and job part length  $q$ , we count the number of times a piece of  $j$  with length  $q$  is scheduled and set  $y_{(q,s_j)}^{(j)}$  accordingly. It is easy to see that the constraints are satisfied.

Now, let  $(x, y)$  be a solution to the MCIP with objective value at most  $m\bar{T} - L$ . We use the solution to construct a schedule: For job  $j$  and configuration  $C$ , we reserve  $x_C^{(j)}$  machines. On each of these machines, we create  $C_h$  slots of length  $h$ , for each module size  $h \in H$ . Note that, because of (9), there is the exact right number of machines for this. Next, consider each job  $j$  and possible job part length  $q$  and create  $y_{(q,s_j)}^{(j)}$  split pieces of length  $q$  and place them together with a setup of  $s_j$  into a slot of length  $s_j + q$  on any machine. Because of (11), the entire job is split up by this, and because of (10), there are enough slots for all the job pieces. Note that the used space in the created schedule is equal to the objective value of  $(x, y)$ , and therefore there is at least  $L$  free space. ◀

**Result.** To assess the running time of the procedure, we mainly need to bound the parameters of the MCIP, namely  $|\mathcal{B}|$ ,  $|H|$ ,  $|\mathcal{M}|$ ,  $|\mathcal{C}|$  and  $D$ . By definition, we have  $|\mathcal{B}| \leq n$  and  $D = 1$ . Since all setup times and job piece lengths are multiples of  $\varepsilon^2 T$  and bounded by  $T$ , we have  $|\mathcal{M}| = \mathcal{O}(1/\varepsilon^4)$  and  $|H| = \mathcal{O}(1/\varepsilon^2)$ . This yields  $|\mathcal{C}| \leq |H|^{\mathcal{O}(1/\varepsilon+2)} = 2^{\mathcal{O}(1/\varepsilon \log 1/\varepsilon)}$ , because the size of each module is at least  $\varepsilon T$  and the size of the configurations bounded by  $(1 + 2\varepsilon)T$ .

According to Observation 1, we now have brick-size  $t = 2^{\mathcal{O}(1/\varepsilon \log 1/\varepsilon)}$ , brick number  $|\mathcal{B}| = n$ ,  $r = |\Gamma| + 1 = \mathcal{O}(1/\varepsilon^2)$  globally uniform and  $s = D = 1$  locally uniform constraints. Because of the scaling step, all occurring numbers in the constraint matrix of the MCIP are bounded by  $1/\varepsilon^2$  and therefore  $\Delta \leq 1/\varepsilon^2$ . Furthermore, each occurring number can be bounded by  $\mathcal{O}(m/\varepsilon^2)$  and this is an upper bound for each variable as well, yielding  $\varphi = \mathcal{O}(\log m/\varepsilon^2)$  and  $\Phi = \mathcal{O}(m/\varepsilon^2)$ . Hence the MCIP, can be solved in time:

$$(rs\Delta)^{\mathcal{O}(r^2s+rs^2)} t^2 n^2 \varphi \log(\Phi) \log(nt\Phi) = 2^{\mathcal{O}(1/\varepsilon^4 \log 1/\varepsilon)} n^2 \log^2 m \log nm$$

When building the actual schedule, we iterate through the jobs and machines, yielding an additional term that is linear in  $n$  and  $m$ . To get the running time stated below Theorem 1, some additional considerations are needed. For details, we refer to the long version of the paper.

### 4.3 Preemptive Model

In the preemptive model, we have to actually consider the time-line of the schedule on each machine instead of just the assignment of the jobs or job pieces, and this causes some difficulties. For instance, we will have to argue that it suffices to look for a schedule with few possible starting points, and we will have to introduce additional constraints in the IP in order to ensure that pieces of the same job do not overlap. Our first step, in dealing with this extra difficulty is to introduce some concepts and notation: For a given schedule with a makespan bound  $T$ , we call a job piece together with its setup a *block*, and we call the schedule  $X$ -layered, for some value  $X$ , if each block starts at a multiple of  $X$ . Corresponding to this, we call the time in the schedule between two directly succeeding multiples of  $X$  a *layer* and the corresponding time on a single machine a *slot*. We number the layers bottom to top and identify them with their number, that is, the set of layers  $\Xi$  is given by  $\{\ell \in \mathbb{Z}_{>0} \mid (\ell - 1)X \leq T\}$ . Note that in an  $X$ -layered schedule, there is at most one block in each slot and for each layer there can be at most one block of each job present. Furthermore, for  $X$ -layered schedules, we slightly alter the definition of free space: We solely count the

space from slots that are completely free. If in such a schedule, for each job there is at most one slot occupied by this job but not fully filled, we additionally call the schedule *layer-compliant*.

**Simplification of the Instance.** In the preemptive model, we distinguish *big*, *medium* and *small* setup jobs, using two parameters  $\delta$  and  $\mu$ : The big setup jobs  $\mathcal{J}^{\text{bst}}$  are those with setup time at least  $\delta T$ , the small  $\mathcal{J}^{\text{sst}}$  have a setup time smaller than  $\mu T$ , and the medium  $\mathcal{J}^{\text{mst}}$  are the ones in between. We set  $\mu = \varepsilon^2 \delta$  and we choose  $\delta \in \{\varepsilon^1, \dots, \varepsilon^{2/\varepsilon^2}\}$  such that the summed up processing time together with the summed up setup time of the medium setup jobs is upper bounded by  $m\varepsilon T$ , i.e.,  $\sum_{j \in \mathcal{J}^{\text{mst}}} (s_j + p_j) \leq m\varepsilon T$ . If there is a schedule with makespan  $T$ , such a choice is possible, because of the pidgeon hole principle, and because the setup time of each job has to occur at least once in any schedule. Similar arguments are widely used, e.g. in the context of geometrical packing algorithms. Furthermore, we distinguish the jobs by processing times, calling those with processing time at least  $\varepsilon T$  *big* and the others *small*. For a given set of jobs  $J$ , we call the subsets of big or small jobs  $J_{\text{big}}$  or  $J_{\text{small}}$  respectively. We perform three simplification steps, aiming for an instance in which the small and medium setup jobs are big; small setup jobs have setup time 0; and for which an  $\varepsilon \delta T$ -layered, layer-compliant schedule exists.

Let  $I_1$  be the instance we get by removing the small jobs with medium setup times  $\mathcal{J}_{\text{small}}^{\text{mst}}$  from the given instance  $I$ .

► **Lemma 8.** *If there is a schedule with makespan at most  $T$  for  $I$ , there is also such a schedule for  $I_1$ , and if there is a schedule with makespan at most  $T'$  for  $I_1$  there is a schedule with makespan at most  $T' + (\varepsilon + \delta)T$  for  $I$ .*

**Proof.** The first claim is obvious. For the second, we create a sequence containing the jobs from  $\mathcal{J}_{\text{small}}^{\text{mst}}$  each directly preceded by its setup time. Recall that the overall length of the objects in this sequence is at most  $m\varepsilon T$ , and the length of each job is bounded by  $\varepsilon T$ . We greedily insert the objects from the sequence, considering each machine in turn. On the current machine we start at time  $T' + \delta T$  and keep inserting until  $T' + \delta T + \varepsilon T$  is reached. If the current object is a setup time, we discard it and continue with the next machine and object. If, on the other hand, it is a job, we split it such that the remaining space on the current machine can be perfectly filled. We can place all objects like this, however the first job part placed on a machine might be missing a setup. We can insert the missing setups because they have length at most  $\delta T$  and between time  $T'$  and  $T' + \delta T$  there is free space. ◀

Next, we consider the jobs with small setup times: Let  $I_2$  be the instance we get by removing the small jobs with small setup times  $\mathcal{J}_{\text{small}}^{\text{sst}}$  and setting the setup time of the big jobs with small setup times to zero, i.e.,  $\bar{s}_j = 0$  for each  $j \in \mathcal{J}_{\text{big}}^{\text{sst}}$ . Note that in the resulting instance each small job has a big setup time. Furthermore, let  $L := \sum_{j \in \mathcal{J}_{\text{small}}^{\text{sst}}} p_j + s_j$ . Then  $L$  is an obvious lower bound for the space taken up by the jobs from  $\mathcal{J}_{\text{small}}^{\text{sst}}$  in any schedule.

► **Lemma 9.** *If there is a schedule with makespan at most  $T$  for  $I_1$ , there is also a  $(T, L)$ -schedule for  $I_2$ ; and if there is a  $\gamma T$ -layered  $(T', L)$ -schedule for  $I_2$ , with  $T'$  a multiple of  $\gamma T$ , there is also a schedule with makespan at most  $(1 + \gamma^{-1}\mu)T' + (\mu + \varepsilon)T$  for  $I_1$ .*

**Proof.** The first claim is obvious, and for the second consider a  $\gamma T$ -layered  $(T', L)$ -schedule for  $I_2$ . We create a sequence that contains the jobs of  $\mathcal{J}_{\text{small}}^{\text{sst}}$  and their setups such that each job is directly preceded by its setup. Remember that the remaining space in partly filled slots is not counted as free space. Hence, since the overall length of the objects in the

sequence is  $L$ , there is enough space in the free slots of the schedule to place them. We do so in a greedy fashion guaranteeing that each job is placed on exactly one machine: We insert the objects from the sequence into the free slots, considering each machine in turn and starting on the current machine from the beginning of the schedule and moving on towards its end. If an object cannot be fully placed into the current slot there are two cases: If its a setup we discard it and if its a job, we cut it and continue placing it in the next slot, or, if the current slot was the last one, we place the rest at the end of the schedule. The resulting schedule is increased by at most  $\varepsilon T$ , which is caused by the last job placed on a machine.

To get a proper schedule for  $I_1$  we have to insert some setup times: For the large jobs with small setup times and for the jobs that were cut in the greedy procedure. We do so by inserting a time window of length  $\mu T$  at each multiple of  $\gamma T$  and at the end of the original schedule on each machine. By this, the schedule is increased by at most  $\gamma^{-1}\mu T' + \mu T$ . Since all the job parts in need of a setup are small and did start at multiples of  $\mu T$  or at the end, we can insert the missing setups. Note that blocks that span over multiple layers are cut by the inserted time windows. This, however, can easily be repaired by moving the cut pieces properly down. ◀

We continue by rounding the medium and big setup and all the processing times. In particular, we round the processing times and the big setup times up to the next multiple of  $\varepsilon\delta T$  and the medium setup times to the next multiple of  $\varepsilon\mu T$ , i.e.,  $\bar{p}_j = \lceil p_j/(\varepsilon\delta T) \rceil \varepsilon\delta T$  for each job  $j$ ,  $\bar{s}_j = \lceil s_j/(\varepsilon\delta T) \rceil \varepsilon\delta T$  for each big setup job  $j \in \mathcal{J}^{\text{bst}}$ , and  $\bar{s}_j = \lceil s_j/(\varepsilon\mu T) \rceil \varepsilon\mu T$  for each medium setup job  $j \in \mathcal{J}_{\text{big}}^{\text{mst}}$ .

► **Lemma 10.** *If there is a  $(T, L)$ -schedule for  $I_2$ , there is also an  $\varepsilon\delta T$ -layered, layer-compliant  $((1 + 3\varepsilon)T, L)$ -schedule for  $I_3$ ; and if there is a  $\gamma T$ -layered  $(T', L)$ -schedule for  $I_3$ , there is also such a schedule for  $I_2$ .*

While the second claim is easy to see, the proof of the first is rather elaborate and can be found in the long version of the paper.

For the big and small setup jobs both processing and setup times are multiples of  $\varepsilon\delta T$ . Therefore, the length of each of their blocks in an  $\varepsilon\delta T$ -layered, layer-compliant schedule is a multiple of  $\varepsilon\delta T$ . For a medium setup job, on the other hand, we know that the overall length of its blocks has the form  $x\varepsilon\delta T + y\varepsilon\mu T$ , with non-negative integers  $x$  and  $y$ . In particular it is a multiple of  $\varepsilon\mu T$ , because  $\varepsilon\delta T = (1/\varepsilon^2)\varepsilon\mu T$ . In a  $\varepsilon\delta T$ -layered, layer-compliant schedule, for each medium setup job the length of all but at most one block is a multiple of  $\varepsilon\delta T$  and therefore a multiple of  $\varepsilon\mu T$ . If both the overall length and the lengths of all but one block are multiples of  $\varepsilon\mu T$ , this is also true for the one remaining block. Hence, we will use the MCIP not to find an  $\varepsilon\delta T$ -layered, layer-compliant schedule in particular, but an  $\varepsilon\delta T$ -layered one with block sizes as described above and maximum free space.

Based on the simplification steps, we define two makespan bounds  $\bar{T}$  and  $\check{T}$ : Let  $\bar{T}$  be the makespan bound we get by the application of the Lemmata 8-10, i.e.,  $\bar{T} = (1 + \mathcal{O}(\varepsilon))T$ . We will use the MCIP to find an  $\varepsilon\delta T$ -layered  $(\bar{T}, L)$ -schedule for  $I_3$ , and apply the Lemmata 8-10 backwards to get schedule for  $I$  with makespan at most  $\check{T} = (1 + \mathcal{O}(\varepsilon))\bar{T} = (1 + \mathcal{O}(\varepsilon))T$ .

**Utilization of the MCIP.** Similar to the splittable case, the basic objects are the (big) jobs, i.e.,  $\mathcal{B} = \mathcal{J}_{\text{big}}$ , and their single value is their processing time ( $D = 1$ ). The modules, on the other hand, are more complicated, because they additionally need to encode which layers are exactly used and, in case of the medium jobs, to which degree the last layer is filled. For the latter, we introduce buffers, representing the unused space in the last layer, and define modules as tuples  $(\ell, q, s, b)$  of starting layer, job piece size, setup time and buffer size. For a



module  $M = (\ell, q, s, b)$ , we write  $\ell_M = \ell$ ,  $q_M = q$ ,  $s_M = s$  and  $b_M = b$ , and we define the size  $\Lambda(M)$  of  $M$  as  $s + q + b$ . The overall set of modules  $\mathcal{M}$  is the union of the modules for big, medium and small setup jobs  $\mathcal{M}^{\text{bst}}$ ,  $\mathcal{M}^{\text{mst}}$  and  $\mathcal{M}^{\text{sst}}$  that are defined in the following. For this let  $Q^{\text{bst}} = \{q \mid q = x\varepsilon\delta T, x \in \mathbb{Z}_{>0}, q \leq \bar{T}\}$  and  $Q^{\text{mst}} = \{q \mid q = x\varepsilon\mu T, x \in \mathbb{Z}_{>0}, q \leq \bar{T}\}$  be the sets of possible job piece sizes of big and medium setup jobs;  $S^{\text{bst}} = \{s \mid s = x\varepsilon\delta T, x \in \mathbb{Z}_{\geq 1/\varepsilon}, s \leq \bar{T}\}$  and  $S^{\text{mst}} = \{s \mid s = x\varepsilon\mu T, x \in \mathbb{Z}_{\geq 1/\varepsilon}, s \leq \delta T\}$  be the sets of possible big and medium setup times;  $B = \{b \mid b = x\varepsilon\mu T, x \in \mathbb{Z}_{\geq 0}, b < \varepsilon\delta T\}$  the set of possible buffer sizes; and  $\Xi = \{1, \dots, 1/(\varepsilon\delta) + 3/\delta\}$  the set of layers. We set:

$$\begin{aligned} \mathcal{M}^{\text{bst}} &= \{(\ell, q, s, 0) \mid \ell \in \Xi, q \in Q^{\text{bst}}, s \in S^{\text{bst}}, (\ell - 1)\varepsilon\delta T + s + q \leq \bar{T}\} \\ \mathcal{M}^{\text{mst}} &= \{(\ell, q, s, b) \in \Xi \times Q^{\text{mst}} \times S^{\text{mst}} \times B \mid x = s + q + b \in \varepsilon\delta T\mathbb{Z}_{>0}, (\ell - 1)\varepsilon\delta T + x \leq \bar{T}\} \\ \mathcal{M}^{\text{sst}} &= \{(\ell, \varepsilon\delta T, 0, 0) \mid \ell \in \Xi\} \end{aligned}$$

Concerning the small setup modules, note that the small setup jobs have a setup time of 0 and therefore may be covered slot by slot. We establish  $\varepsilon\mu T = 1$  via scaling, to ensure integral values. A big, medium or small job is eligible for a module, if it is also big, medium or small respectively and the setup times fit.

We have to avoid that two modules  $M_1, M_2$ , whose corresponding time intervals overlap, are used to cover the same job or in the same configuration. Such an overlap is given, if there is some layer  $\ell$  used by both of them, that is,  $(\ell_{M_1} - 1)\varepsilon\delta T \leq (\ell - 1)\varepsilon\delta T < (\ell_{M_2} - 1)\varepsilon\delta T + \Lambda(M_2)$  for both  $M \in \{M_1, M_2\}$ . Hence, for each layer  $\ell \in \Xi$ , we set  $\mathcal{M}_\ell \subseteq \mathcal{M}$  to be the set of modules that use layer  $\ell$ . Furthermore, we partition the modules into groups  $\Gamma$  by size and starting layer, i.e.,  $\Gamma = \{G \subseteq \mathcal{M} \mid M, M' \in G \Rightarrow \Lambda(M) = \Lambda(M') \wedge \ell_M = \ell_{M'}\}$ . The size of a group  $G \in \Gamma$  is the size of a module from  $G$ , i.e.  $\Lambda(G) = \Lambda(M)$  for  $M \in G$ . Unlike before, we consider *configurations of module groups* rather than module sizes. More precisely, the set of configurations  $\mathcal{C}$  is given by the configurations of groups, such that for each layer at most one group using this layer is chosen, i.e.,  $\mathcal{C} = \{C \in \mathbb{Z}_{\geq 0}^\Gamma \mid \forall \ell \in \Xi : \sum_{G \subseteq \mathcal{M}_\ell} C_G \leq 1\}$ . With this definition, we prevent overlap conflicts on the machines. Note that unlike in the cases considered so far, the size of a configuration does not correspond to a makespan in the schedule, but to used space, and the makespan bound is realized in the definition of the modules instead of in the definition of the configurations. To also avoid conflicts for the jobs, we extend the basic MCIP with additional locally uniform constraints. In particular, the constraints of the extended MCIP for the above definitions with adapted notation and without duplication of the configuration variables are given by:

$$\sum_{C \in \mathcal{C}} x_C = m \tag{12}$$

$$\sum_{C \in \mathcal{C}(T)} C_G x_C = \sum_{j \in \mathcal{J}} \sum_{M \in G} y_M^{(j)} \quad \forall G \in \Gamma \tag{13}$$

$$\sum_{M \in \mathcal{M}} q_M y_M^{(j)} = p_j \quad \forall j \in \mathcal{J} \tag{14}$$

$$\sum_{M \in \mathcal{M}_\ell} y_M^{(j)} \leq 1 \quad \forall j \in \mathcal{J}, \ell \in \Xi \tag{15}$$

Like in the first two cases, we minimize the summed up size of the configurations, via the objective function  $\sum_C \Lambda(C)x_C$ . Note that in this case the size of a configuration does not have to equal its height. It is easy to see that the last constraint is indeed locally uniform. However, since we have an inequality instead of an equality, we have to introduce  $|\Xi|$  slack variables in each brick, yielding:

► **Observation 2.** *The MCIP extended like above is an  $n$ -fold IP with brick-size  $t = |\mathcal{M}| + |\mathcal{C}| + |\Xi|$ , brick number  $n = |\mathcal{J}|$ ,  $r = |\Gamma| + 1$  globally uniform and  $s = D + |\Xi|$  locally uniform constraints.*

► **Lemma 11.** *With the above definitions, there is an  $\varepsilon\delta T$ -layered  $(\bar{T}, L)$ -schedule for  $I_3$  in which the length of a block is a multiple of  $\varepsilon\delta T$ , if it belongs to a small or big setup job, or a multiple of  $\varepsilon\mu T$  otherwise, iff the extended MCIP has a solution with objective value at most  $m\bar{T} - L$ .*

**Proof.** We first consider such a schedule for  $I_3$ . For each machine, we can derive a configuration that is given by the starting layers of the blocks together with the summed up length of the slots the respective block is scheduled in. The size of the configuration  $C$  is equal to the used space on the respective machine. Hence, we can fix some arbitrary job  $j$  and set  $x_C^{(j)}$  to the number of machines corresponding to  $j$  (and  $x_C^{(j')} = 0$  for  $j' \neq j$ ). Keeping in mind that in an  $\varepsilon\delta T$ -layered schedule the free space is given by the free slots, the above definition yields an objective value bounded by  $m\bar{T} - L$ , because there was free space of at least  $L$ . Next, we consider the module variables for each job  $j$  in turn: If  $j$  is a small setup job, we set  $y_{(\ell, \varepsilon\delta T, 0, 0)}^{(j)}$  to 1, if the  $j$  occurs in  $\ell$ , and to 0 otherwise. Now, let  $j$  be a big setup job. For each of its blocks, we set  $y_{(\ell, z-s_j, s_j, 0)}^{(j)} = 1$ , where  $\ell$  is the starting layer and  $z$  the length of the block. The remaining variables are set to 0. Lastly, let  $j$  be a medium setup job. For each of its blocks, we set  $y_{(\ell, z-s_j, s_j, b)}^{(j)} = 1$ , where  $\ell$  is the starting layer of the block,  $z$  its length and  $b = \lceil z/(\varepsilon\delta T) \rceil \varepsilon\delta T - z$ . Again, the remaining variables are set to 0. It is easy to verify that all constraints are satisfied by this solution.

If, on the other hand, we have a solution  $(x, y)$  to the MCIP with objective value at most  $m\bar{T} - L$ , we reserve  $\sum_j x_C^{(j)}$  machines for each configuration  $C$ . There are enough machines to do this, because of (12). On each of these machines we reserve space: For each  $G \in \Gamma$ , we create an allocated space of length  $\Lambda(G)$  starting from the starting layer of  $G$ , if  $C_G = 1$ . Let  $j$  be a job and  $\ell$  be a layer. If  $j$  has a small setup time, we create  $y_{(\ell, \varepsilon\delta T, 0, 0)}^{(j)}$  pieces of length  $\varepsilon\delta T$  and place these pieces into allocated spaces of length  $\varepsilon\delta T$  in layer  $\ell$ . If, on the other hand,  $j$  is a big or medium setup job, we consider each possible job part length  $q \in Q^{\text{bst}}$  or  $q \in Q^{\text{mst}}$ , create  $y_{(\ell, q, s_j, 0)}^{(j)}$  or  $y_{(\ell, q, s_j, b)}^{(j)}$ , with  $b = \lceil q/(\varepsilon\delta T) \rceil \varepsilon\delta T - q$ , pieces of length  $q$ , and place them together with their setup time into allocated spaces of length  $q$  in layer  $\ell$ . Because of (14) the entire job is split up by this, and because of (13) there are enough allocated spaces for all the job pieces. The makespan bound is ensured by the definition of the modules, and overlaps are avoided, due to the definition of the configurations and (15). Furthermore, the used slots have an overall length equal to the objective value of  $(x, y)$  and therefore there is at least  $L$  free space. ◀

**Result.** We analyze the running time of the procedure, and start by bounding the parameters of the extended MCIP. We have  $|\mathcal{B}| = n$  and  $D = 1$  by definition, and the number of layers  $|\Xi|$  is obviously  $\mathcal{O}(1/(\varepsilon\delta)) = \mathcal{O}(1/\varepsilon^{2/\varepsilon+1}) = 2^{\mathcal{O}(1/\varepsilon \log 1/\varepsilon)}$ . Furthermore, it is easy to see that  $|Q^{\text{bst}}| = \mathcal{O}(1/(\varepsilon\delta))$ ,  $|Q^{\text{mst}}| = \mathcal{O}(1/(\varepsilon^3\delta))$ ,  $|S^{\text{bst}}| = \mathcal{O}(1/(\varepsilon\delta))$ ,  $|S^{\text{mst}}| = \mathcal{O}(1/\varepsilon^3)$ , and  $|B| = \mathcal{O}(1/\varepsilon^2)$ . This gives us  $\mathcal{M}^{\text{bst}} \leq |\Xi||Q^{\text{bst}}||S^{\text{bst}}|$ ,  $\mathcal{M}^{\text{mst}} \leq |\Xi||Q^{\text{mst}}||S^{\text{mst}}||B|$  and  $\mathcal{M}^{\text{sst}} = |\Xi|$ , and therefore  $|\mathcal{M}| = |\mathcal{M}^{\text{bst}}| + |\mathcal{M}^{\text{mst}}| + |\mathcal{M}^{\text{sst}}| = 2^{\mathcal{O}(1/\varepsilon \log 1/\varepsilon)}$ . Since their are  $\mathcal{O}(1/(\delta\varepsilon))$  distinct module sizes, the number of groups  $|\Gamma|$  can be bounded by  $\mathcal{O}(|\Xi|/(\varepsilon\delta)) = 2^{\mathcal{O}(1/\varepsilon \log 1/\varepsilon)}$ . Hence, for the number of configurations we get  $|\mathcal{C}| = \mathcal{O}((1/(\varepsilon\delta))^{|\Gamma|}) = 2^{\mathcal{O}(1/\varepsilon \log 1/\varepsilon)}$ . By Observation 2, the modified MCIP has  $r = 2^{\mathcal{O}(1/\varepsilon \log 1/\varepsilon)}$  many globally and  $s = 2^{\mathcal{O}(1/\varepsilon \log 1/\varepsilon)}$  many locally uniform constraints; its brick number is  $n$ , and its brick size is  $t = 2^{\mathcal{O}(1/\varepsilon \log 1/\varepsilon)}$ . All occurring values in the matrix are bounded by  $\bar{T}$ , yielding  $\Delta \leq \bar{T} = 1/(\varepsilon\mu) + 1/\mu =$

$2^{\mathcal{O}(1/\varepsilon \log 1/\varepsilon)}$ , due to the scaling step. Furthermore, the numbers in the input can be bounded by  $m2^{\mathcal{O}(1/\varepsilon \log 1/\varepsilon)}$  and all variables can be upper bounded by  $\mathcal{O}(m)$ . Hence, we have  $\varphi = \mathcal{O}(\log m + 1/\varepsilon \log 1/\varepsilon)$  and  $\Phi = \mathcal{O}(m)$ , and due to Theorem 3 we can solve the MCIP in time:

$$(rs\Delta)^{\mathcal{O}(r^2s+rs^2)} t^2 n^2 \varphi \log(\Phi) \log(nt\Phi) = 2^{2^{\mathcal{O}(1/\varepsilon \log 1/\varepsilon)}} n^2 \log^2 m \log nm$$

When building the actual schedule we iterate through the jobs and machines yielding the slightly increased running time stated below Theorem 1.

## 5 Conclusion

We presented a more advanced version of the classical configuration IP, showed that it can be solved efficiently using algorithms for  $n$ -fold IPs, and developed techniques to employ the new IP for the formulation of efficient polynomial time approximation schemes for three scheduling problems with setup times, for which no such algorithms were known before.

For further research the immediate questions are whether improved running times for the considered problems, in particular for the preemptive model, can be achieved; whether the MCIP can be solved more efficiently; and to which other problems it can be reasonably employed. From a broader perspective, it would be interesting to further study the potential of new algorithmic approaches in integer programming for approximation, and, on the other hand, further study the respective techniques themselves.

---

## References

- 1 Ali Allahverdi, Jatinder ND Gupta, and Tariq Aldowaisan. A review of scheduling research involving setup considerations. *Omega*, 27(2):219–239, 1999.
- 2 Ali Allahverdi, CT Ng, TC Edwin Cheng, and Mikhail Y Kovalyov. A survey of scheduling problems with setup times or costs. *European journal of operational research*, 187(3):985–1032, 2008.
- 3 Noga Alon, Yossi Azar, Gerhard J Woeginger, and Tal Yadid. Approximation schemes for scheduling on parallel machines. *Journal of Scheduling*, 1(1):55–66, 1998.
- 4 Bo Chen. A better heuristic for preemptive parallel machine scheduling with batch setup times. *SIAM Journal on Computing*, 22(6):1303–1318, 1993.
- 5 Bo Chen, Yinyu Ye, and Jiawei Zhang. Lot-sizing scheduling with batch setup times. *Journal of Scheduling*, 9(3):299–310, 2006.
- 6 Lin Chen, Dániel Marx, Deshi Ye, and Guochuan Zhang. Parameterized and approximation results for scheduling with a low rank processing time matrix. In *LIPICs-Leibniz International Proceedings in Informatics*, volume 66. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2017.
- 7 José Correa, Alberto Marchetti-Spaccamela, Jannik Matuschke, Leen Stougie, Ola Svensson, Víctor Verdugo, and José Verschae. Strong LP formulations for scheduling splittable jobs on unrelated machines. *Mathematical Programming*, 154(1-2):305–328, 2015.
- 8 José Correa, Victor Verdugo, and José Verschae. Splitting versus setup trade-offs for scheduling to minimize weighted completion time. *Operations Research Letters*, 44(4):469–473, 2016.
- 9 Friedrich Eisenbrand, Christoph Hunkenschröder, and Kim-Manuel Klein. Faster Algorithms for Integer Programs with Block Structure. In *45th International Colloquium on Automata, Languages, and Programming, ICALP 2018, July 9-13, 2018, Prague, Czech Republic*, pages 49:1–49:13, 2018.

- 10 Paul C Gilmore and Ralph E Gomory. A linear programming approach to the cutting-stock problem. *Operations research*, 9(6):849–859, 1961.
- 11 Michel X Goemans and Thomas Rothvoß. Polynomiality for bin packing with a constant number of item types. In *Proceedings of the twenty-fifth annual ACM-SIAM symposium on Discrete algorithms*, pages 830–839. SIAM, 2014.
- 12 Raymond Hemmecke, Shmuel Onn, and Lyubov Romanchuk. N-fold integer programming in cubic time. *Mathematical Programming*, pages 1–17, 2013.
- 13 Dorit S Hochbaum and David B Shmoys. Using dual approximation algorithms for scheduling problems theoretical and practical results. *Journal of the ACM (JACM)*, 34(1):144–162, 1987.
- 14 Klaus Jansen, Kim-Manuel Klein, Marten Maack, and Malin Rau. Empowering the Configuration-IP – New PTAS Results for Scheduling with Setup Times. *CoRR*, abs/1801.06460v2, 2018. [arXiv:1801.06460](https://arxiv.org/abs/1801.06460).
- 15 Klaus Jansen, Kim-Manuel Klein, and José Verschae. Closing the Gap for Makespan Scheduling via Sparsification Techniques. In *43rd International Colloquium on Automata, Languages, and Programming, ICALP 2016, July 11-15, 2016, Rome, Italy*, pages 72:1–72:13, 2016.
- 16 Klaus Jansen and Felix Land. Non-preemptive Scheduling with Setup Times: A PTAS. In *European Conference on Parallel Processing*, pages 159–170. Springer, 2016.
- 17 Ravi Kannan. Minkowski’s convex body theorem and integer programming. *Mathematics of operations research*, 12(3):415–440, 1987.
- 18 Dusan Knop and Martin Koutecký. Scheduling meets n-fold Integer Programming. *Journal of Scheduling*, pages 1–11, 2017.
- 19 Dusan Knop, Martin Koutecký, and Matthias Mnich. Combinatorial n-fold Integer Programming and Applications. In *25th Annual European Symposium on Algorithms, ESA 2017, September 4-6, 2017, Vienna, Austria*, pages 54:1–54:14, 2017.
- 20 Martin Koutecký, Asaf Levin, and Shmuel Onn. A Parameterized Strongly Polynomial Algorithm for Block Structured Integer Programs. In *45th International Colloquium on Automata, Languages, and Programming, ICALP 2018, July 9-13, 2018, Prague, Czech Republic*, pages 85:1–85:14, 2018.
- 21 Hendrik W Lenstra Jr. Integer programming with a fixed number of variables. *Mathematics of operations research*, 8(4):538–548, 1983.
- 22 Alexander Mäcker, Manuel Malatyali, Friedhelm Meyer auf der Heide, and Sören Riechers. Non-preemptive scheduling on machines with setup times. In *Workshop on Algorithms and Data Structures*, pages 542–553. Springer, 2015.
- 23 Clyde L Monma and Chris N Potts. Analysis of heuristics for preemptive parallel machine scheduling with batch setup times. *Operations Research*, 41(5):981–993, 1993.
- 24 Shmuel Onn. Nonlinear discrete optimization. *Zurich Lectures in Advanced Mathematics, European Mathematical Society*, 2010.
- 25 Frans Schalekamp, René Sitters, Suzanne Van Der Ster, Leen Stougie, Víctor Verdugo, and Anke Van Zuylen. Split scheduling with uniform setup times. *Journal of scheduling*, 18(2):119–129, 2015.
- 26 Petra Schuurman and Gerhard J Woeginger. Preemptive scheduling with job-dependent setup times. In *Proceedings of the tenth annual ACM-SIAM symposium on Discrete algorithms*, pages 759–767. Society for Industrial and Applied Mathematics, 1999.



# Being Corrupt Requires Being Clever, But Detecting Corruption Doesn't

Yan Jin<sup>1</sup>

MIT, 77 Massachusetts Ave, MA, USA  
yjin1@mit.edu

Elchanan Mossel<sup>2</sup>

MIT, 77 Massachusetts Ave, MA, USA  
elmos@mit.edu

Govind Ramnarayan<sup>3</sup>

MIT, 77 Massachusetts Ave, MA, USA  
govind@mit.edu

---

## Abstract

We consider a variation of the problem of corruption detection on networks posed by Alon, Mossel, and Pemantle '15. In this model, each vertex of a graph can be either truthful or corrupt. Each vertex reports about the types (truthful or corrupt) of all its neighbors to a central agency, where truthful nodes report the true types they see and corrupt nodes report adversarially. The central agency aggregates these reports and attempts to find a single truthful node. Inspired by real auditing networks, we pose our problem for arbitrary graphs and consider corruption through a computational lens. We identify a key combinatorial parameter of the graph  $m(G)$ , which is the minimal number of corrupted agents needed to prevent the central agency from identifying a single corrupt node. We give an efficient (in fact, linear time) algorithm for the central agency to identify a truthful node that is successful whenever the number of corrupt nodes is less than  $m(G)/2$ . On the other hand, we prove that for any constant  $\alpha > 1$ , it is NP-hard to find a subset of nodes  $S$  in  $G$  such that corrupting  $S$  prevents the central agency from finding one truthful node and  $|S| \leq \alpha m(G)$ , assuming the Small Set Expansion Hypothesis (Raghavendra and Steurer, STOC '10). We conclude that being corrupt requires being clever, while detecting corruption does not.

Our main technical insight is a relation between the minimum number of corrupt nodes required to hide all truthful nodes and a certain notion of vertex separability for the underlying graph. Additionally, this insight lets us design an efficient algorithm for a corrupt party to decide which graphs require the fewest corrupted nodes, up to a multiplicative factor of  $O(\log n)$ .

**2012 ACM Subject Classification** Theory of computation → Problems, reductions and completeness

**Keywords and phrases** Corruption detection, PMC Model, Small Set Expansion, Hardness of Approximation

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2019.45

**Related Version** A full version of the paper with all proofs can be found at [8], <https://arxiv.org/abs/1809.10325>.

---

<sup>1</sup> Partially supported by Institute for Data, Systems and Society Fellowship.

<sup>2</sup> Partially supported by awards ONR N00014-16-1-2227, NSF CCF1665252 and DMS-1737944.

<sup>3</sup> Partially supported by awards NSF CCF 1665252 and DMS-1737944.



© Yan Jin, Elchanan Mossel, and Govind Ramnarayan;  
licensed under Creative Commons License CC-BY

10th Innovations in Theoretical Computer Science (ITCS 2019).

Editor: Avrim Blum; Article No. 45; pp. 45:1–45:14



Leibniz International Proceedings in Informatics

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

**Acknowledgements** We would like to thank Paxton Turner, Vishesh Jain, and Pasin Manurangsi for useful discussions and pointers to resources.

## 1 Introduction

### 1.1 Corruption Detection and Problem Set-up

We study the problem of identifying truthful nodes in networks, in the model of *corruption detection on networks* posed by Alon, Mossel, and Pemantle [1]. In this model, we have a network represented by a (possibly directed) graph. Nodes can be *truthful* or *corrupt*. Each node audits its outgoing neighbors to see whether they are truthful or corrupt, and sends reports of their identities to a central agency. The central agent, who is not part of the graph, aggregates the reports and uses them to identify truthful and corrupt nodes. Truthful nodes report truthfully (and correctly) on their neighbors, while corrupt nodes have no such restriction: they can assign arbitrary reports to their neighbors, regardless of whether their neighbors are truthful or corrupt, and coordinate their efforts with each other to prevent the central agency from gathering useful information.

In [1], the authors consider the problem of recovering the identities of almost all nodes in a network in the presence of many corrupt nodes; specifically, when the fraction of corrupt nodes can be very close to  $1/2$ . They call this the *corruption detection* problem. They show that the central agency can recover the identity of most nodes correctly even in certain bounded-degree graphs, as long as the underlying graph is a sufficiently good expander. The required expansion properties are known to hold for a random graph or Ramanujan graph of sufficiently large (but constant) degree, which yields undirected graphs that are amenable to corruption detection. Furthermore, they show that some level of expansion is necessary for identifying truthful nodes, by demonstrating that the corrupt nodes can stop the central agency from identifying any truthful node when the graph is a very bad expander (e.g. a cycle), even if the corrupt nodes only make up 0.01 fraction of the network.

This establishes that very good expanders are very good for corruption detection, and very bad expanders can be very bad for corruption detection. We note that this begs the question of how effective graphs that do not fall in either of these categories are for corruption detection. In the setting of [1], we could ask the following: given an *arbitrary* undirected graph, what is the smallest number of corrupt nodes that can prevent the identification of almost all nodes? When there are fewer than this number, can the central agency *efficiently* identify almost all nodes correctly? Alon, Mossel, and Pemantle study these questions for the special cases of highly expanding graphs and poorly expanding graphs, but do not address general graphs.

Additionally, [1] considers corruption detection when the corrupt agencies can choose their locations and collude arbitrarily, with no bound on their computational complexity. This is perhaps overly pessimistic: after all, it is highly unlikely that corrupt agencies can solve NP-hard problems efficiently and if they can, thwarting their covert operations is unlikely to stop their world domination. We suggest a model that takes into account computational considerations, by factoring in the computation time required to select the nodes in a graph that a corrupt party chooses to control. This yields the following question from the viewpoint of a corrupt party: given a graph, can a corrupt party compute the smallest set of nodes it needs to corrupt *in polynomial time*?

In addition to being natural from a mathematical standpoint, these questions are also well-motivated socially. It would be naïve to assert that we can weed out corruption in the real world by simply designing auditing networks that are expanders. Rather, these



networks may already be formed, and infeasible to change in a drastic way. Given this, we are less concerned with finding certain graphs that are good for corruption detection, but rather discerning how good *existing* graphs are; specifically, how many corrupt nodes they can tolerate. In particular, since the network structure could be out of the control of the central agency, algorithms for the central agency to detect corruption on arbitrary graphs seem particularly important.

It is also useful for the *corrupt* agency to have an algorithm with guarantees for any graph. Consider the following example of a corruption detection problem from the viewpoint of a corrupt organization. Country A wants to influence policy in country B, and wants to figure out the most efficient way to place corrupted nodes within country B to make this happen. However, if the central government of B can confidently identify truthful nodes, they can weight those nodes' opinions more highly, and thwart country A's plans. Hence, the question country A wants to solve is the following: given the graph of country B, can country A compute the optimal placement of corrupt nodes to prevent country B from finding truthful nodes? We note that in this question, too, the graph of country B is fixed, and hence, country A would like to have an algorithm that takes as input *any* graph and computes the optimal way to place corrupt nodes in order to hide all the truthful nodes.

We study the questions above for a variant of the corruption detection problem in [1], in which the goal of the central agency is to find a single truthful node. While this goal is less ambitious than the goal of identifying almost all the nodes, we think it is a very natural question in the context of corruption. For one, if the central agency can find a single truthful node, they can use the trusted reports from that node to identify more truthful and corrupt nodes that it might be connected to. The central agency may additionally weight the opinions of the truthful nodes more when making policy decisions (as alluded to in the example above), and can also incentivize truthfulness by rewarding truthful nodes that it finds and giving them more influence in future networks if possible (by increasing their out-degrees). Moreover, our proofs and results extend to finding larger number of truthful nodes as we discuss below.

Our results stem from a tie between the problem of finding a single truthful node in a graph and a measure of vertex separability of the graph. This tie not only yields an efficient and relatively effective algorithm for the central agency to find a truthful node, but also allows us to relate corrupt party's strategy to the problem of finding a good vertex separator for the graph. Hence, by analyzing the purely graph-theoretic problem of finding a good vertex separator, we can characterize the difficulty of finding a good set of nodes to corrupt. Similar notions of vertex separability have been studied previously (e.g. [13, 17, 3]), and we prove NP-hardness for the notion relevant to us assuming the Small Set Expansion Hypothesis (SSEH). The *Small Set Expansion Hypothesis* is a hypothesis posed by Raghavendra and Steurer [19] that is closely related to the famous Unique Games Conjecture of Khot [11]. In fact, [19] shows that the SSEH implies the Unique Games Conjecture. The SSEH yields hardness results that are not known to follow directly from the UGC, especially for graph problems like sparsest cut and treewidth ([20] and [2] respectively), among others.

## 1.2 Our Results

We now outline our results more formally. We analyze the variant of corruption detection where the central agency's goal is to find a single truthful node. First, we study how effectively the central agency can identify a truthful node on an arbitrary graph, given a set of reports.

Given an undirected graph<sup>4</sup>  $G$ , we let  $m(G)$  denote the minimal number of corrupted nodes required to stop the central agency from finding a truthful node, where the minimum is taken over all strategies of the corrupt party (not just computationally bounded ones). We informally call  $m(G)$  the “critical” number of corrupt nodes for a graph  $G$ . Then, we show the following:

► **Theorem 1.** *Fix a graph  $G$  and suppose that the corrupt party has a budget  $b \leq m(G)/2$ . Then the central agency can identify a truthful node, regardless of the strategy of the corrupt party, and without knowledge of either  $m(G)$  or  $b$ . Furthermore, the central agency's algorithm runs in linear time (in the number of edges in the graph  $G$ ).*

Next, we consider the question from the viewpoint of the corrupt party: can the corrupt party efficiently compute the most economical way to allocate nodes to prevent the central agency from finding a truthful node? Concretely, we focus on a natural decision version of the question: given a graph  $G$  and an upper bound on the number of possible corrupted nodes  $k$ , can the corrupt party prevent the central agency from finding a truthful node?

We actually focus on an easier question: can the corrupt party accurately compute  $m(G)$ , the minimum number of nodes that they need to control to prevent the central agency from finding a truthful node? Not only do we give evidence that computing  $m(G)$  exactly is computationally hard, but we also provide evidence that  $m(G)$  is hard to approximate. Specifically, we show that approximating  $m(G)$  to any constant factor is NP-hard under the Small Set Expansion Hypothesis (SSEH); or in other words, that it is SSE-hard.

► **Theorem 2.** *For every  $\beta > 1$ , there is a constant  $\epsilon > 0$  such that the following is true. Given a graph  $G = (V, E)$ , it is SSE-hard to distinguish between the case where  $m(G) \leq \epsilon \cdot |V|$  and  $m(G) \geq \beta \cdot \epsilon \cdot |V|$ . Or in other words, the problem of approximating the critical number of corrupt nodes for a graph to within any constant factor is SSE-hard.*

This Theorem immediately implies the following Corollary 25.

► **Corollary 25 (restated).** Assume the SSE Hypothesis and that  $P \neq NP$ . Fix any  $\beta > 1$ . There does not exist a polynomial-time algorithm that takes as input an arbitrary graph  $G = (V, E)$  and outputs a set of nodes  $S$  with size  $|S| \leq O(\beta \cdot m(G))$ , such that corrupting  $S$  prevents the central agency from finding a truthful node.

We note that in Corollary 25, the bad party's input is only the graph  $G$ : specifically, they do not have knowledge about the value of  $m(G)$ .

Our proof for Theorem 2 is similar to the proof of Austrin, Pitassi, and Wu [2] for the SSE-hardness of approximating treewidth. This is not a coincidence: in fact, the “soundness” in their reduction involves proving that their graph does not have a good  $1/2$  vertex separator, where the notion of vertex separability (from [4]) is very related to the version we use to categorize the problem of hiding a truthful vertex. We give the proof of Theorem 2 in Section 3.2.

However, if one allows for an approximation factor of  $O(\log |V|)$ , then  $m(G)$  can be approximated efficiently. Furthermore, this yields an approximation algorithm that the corrupt party can use to find a placement that hinders detection of a truthful node.

► **Theorem 3.** *There is a polynomial-time algorithm that takes as input a graph  $G = (V, E)$  and outputs a set of nodes  $S$  with size  $|S| \leq O(\log |V| \cdot m(G))$ , such that corrupting  $S$  prevents the central agency from finding a truthful node.*

---

<sup>4</sup> Unless explicitly specified, all graphs are undirected by default.

The proof of Theorem 3, given in Section 3.2, uses a bi-criterion approximation algorithm for the  $k$ -vertex separator problem given by [13]. As alluded to in Section 1.1, Theorems 2 and 3 both rely on an approximate characterization of  $m(G)$  in terms of a measure of vertex separability of the graph  $G$ , which we give in Section 3.

Additionally, we note that we can adapt Theorems 1 and 2 (as well as Corollary 25) to a more general setting, where the central agency wants to recover some arbitrary number of truthful nodes, where the number of nodes can be proportional to the size of the graph. We describe how to modify our proofs to match this more general setting in Section 5 in full-length version [8].

Together, Theorems 1 and 2 uncover a surprisingly positive result for corruption detection: it is computationally easy for the central agency to find a truthful node when the number of corrupted nodes is only somewhat smaller than the “critical” number for the underlying graph, but it is in general computationally hard for the corrupt party to hide the truthful nodes even when they have a budget that far exceeds the “critical” number for the graph.

### 1.2.1 Results for Directed Graphs

As noted in [1], it is unlikely that real-world auditing networks are undirected. For example, it is likely that the FBI has the authority to audit the Cambridge police department, but it is also likely that the reverse is untrue. Therefore, we would like the central agency to be able to find truthful nodes in directed graphs in addition to undirected graphs. We notice that the algorithm we give in Theorem 1 extends naturally to directed graphs.

► **Theorem 4.** *Fix a directed graph  $D$  and suppose that the corrupt party has a budget  $b \leq m(D)/2$ . Then the central agency can identify a truthful node, regardless of the strategy of the corrupt party, and without the knowledge of either  $m(D)$  or  $b$ . Furthermore, the central agency’s algorithm runs in linear time.*

The proof of Theorem 4 is similar to the proof of Theorem 1, and effectively relates the problem of finding a truthful node on directed graphs to a similar notion of vertex separability, suitably generalized to directed graphs.

### 1.2.2 Results for Finding An Arbitrary Number of Good Nodes

In fact, the problem of finding one good node is just a special case of finding an arbitrary number of good nodes,  $g$ , on the graph  $G$ . We define  $m(G, g)$  as the minimal number of bad nodes required to prevent the identification of  $g$  good nodes on the graph  $G$ . We relate it to an analogous vertex separation notion, and prove the following two theorems, which are extensions of Theorems 1 and 2 to this setting.

► **Theorem 5.** *Fix a graph  $G$  and the number of good nodes to recover,  $g$ . Suppose that the corrupt party has a budget  $b \leq m(G, g)/2$ . If  $g < |V| - 2b$ , then the central agency can identify  $g$  truthful nodes, regardless of the strategy of the corrupt party, and without knowledge either of  $m(G, g)$  or  $b$ . Furthermore, the central agency’s algorithm runs in linear time.*

► **Theorem 6.** *For every  $\beta > 1$  and every  $0 < \delta < 1$ , there is a constant  $\epsilon > 0$  such that the following is true. Given a graph  $G = (V, E)$ , it is SSE-hard to distinguish between the case where  $m(G, \delta|V|) \leq \epsilon \cdot |V|$  and  $m(G, \delta|V|) \geq \beta \cdot \epsilon \cdot |V|$ . Or in other words, the problem of approximating the critical number of corrupt nodes such that it is impossible to find  $\delta|V|$  good nodes within any constant factor is SSE-hard.*

The proof of Theorem 6 is similar to the proof of Theorem 1, and the hardness of approximation proof also relies on the same graph reduction and SSE conjecture. Proofs are presented in Section 5 in our full-length paper [8].

### 1.3 Related Work

The model of corruptions posed by [1] is identical to a model first suggested by Preparata, Metze, and Chien [18], who introduced the model in the context of detecting failed components in digital systems. This work (as well as many follow-ups, e.g. [9, 12]) looked at the problem of characterizing which networks can detect a certain number of corrupted nodes. Xu and Huang [22] give necessary and sufficient conditions for identifying a single corrupted node in a graph, although their characterization is not algorithmically efficient. There are many other works on variants of this problem (e.g. [21, 5]), including recovering node identities with one-sided or two-sided error probabilities in the local reports [14] and adaptively finding truthful nodes [7].

We note that our model of a computationally bounded corrupt party and our stipulation that the graph is fixed ahead of time rather than designed by the central agency, which are our main contributions to the model, seem more naturally motivated in the setting of corruptions than in the setting of designing digital systems. Even the question of identifying a single truthful node could be viewed as more naturally motivated in the setting of corruptions than in the setting of diagnosing systems. We believe there are likely more interesting theoretical questions to be discovered by approaching the PMC model through a corruptions lens.

The identifiability of a single node in the corruptions setting was studied in a recent paper of Mukwembi and Mukwembi [15]. They give a linear time greedy algorithm to recover the identify of a single node in many graphs, *provided that corrupt nodes always report other corrupt nodes as truthful*. Furthermore, this assumption allows them to reduce identifying all nodes to identifying a single node. They argue that such an assumption is natural in the context of corruptions, where corrupt nodes are selfishly incentivized not to out each other. However, in our setting, corrupt nodes can not only betray each other, but are in fact incentivized to do so for the good of the overarching goal of the corrupt party (to prevent the central agency from identifying a truthful node). Given [15], it is not a surprise that the near-optimal strategies we describe for the corrupt party in this paper crucially rely on the fact that the nodes can report each other as corrupt.

Our problem of choosing the best subset of nodes to corrupt bears intriguing similarities to the problem of influence maximization studied by [10], where the goal is to find an optimal set of nodes to target in order to maximize the adoption of a certain technology or product. It is an interesting question to see if there are further similarities between these two areas. Additionally, social scientists have studied corruption extensively (e.g. [6], [16]), though to the best of our knowledge they have not studied it in the graph-theoretic way that we do in this paper.

### 1.4 Comparison to Corruption in Practice

Finally, we must address the elephant in the room. Despite our theoretical results, corruption *is* prevalent in many real-world networks, and yet in many scenarios it is not easy to pinpoint even a single truthful node. One reason for that is that some of assumptions do not seem to hold in some real world networks. For example, we assume that audits from the truthful nodes are not only non-malicious, but also perfectly reliable. In practice this assumption is unlikely to be true: many truthful nodes could be non-malicious but simply unable to audit

their neighbors accurately. Further assumptions that may not hold in some scenarios include the notion of a central agency that is both uncorrupted and has access to reports from every agency, and possibly even the assumption that the number of corrupt nodes is less than  $|V|/2$ . In addition, networks  $G$  may have very low critical numbers  $m(G)$  in practice. For example, there could be a small set of three nodes (named, “President”, “Congress” and “Houses”) that is all corrupt, and all audits in the graph are performed by one of these three nodes. It is thus plausible that a corrupt party could use the structure of realistic auditing networks for their corruption strategy to overcome our worst-case hardness result.

While this points to some shortcomings of our model, it also points out ways to change policy that would potentially bring the real world closer to our idealistic scenario, where a corrupt party has a much more difficult computational task than the central agency. For example, we can speculate that perhaps information should be gathered by a transparent centralized agency, that significant resources should go into ensuring that the centralized agency is not corrupt, and that networks ought to have good auditing structure (without important agencies that can be audited by very few nodes).

## 2 Preliminaries

### 2.1 General Preliminaries

We denote undirected graphs by  $G = (V, E)$ , where  $V$  is the vertex set of the graph and  $E$  is the edge set. We denote directed graphs by  $D = (V, E_D)$ . When the underlying graph is clear, we may drop the subscripts. Given a vertex  $u$  in an undirected graph  $G$ , we let  $\mathcal{N}(u)$  denote the *neighborhood* (set of neighbors) of the vertex in  $G$ . Similarly, given a vertex  $u$  in a directed graph  $D$ , let  $\mathcal{N}(u)$  denote the set of *outgoing* neighbors of  $u$ : that is, vertices  $v \in V$  such that  $(u, v) \in E_D$ .

#### 2.1.1 Vertex Separator

► **Definition 7** (*k*-vertex separator, [17], [3]). For any  $k \geq 0$ , we say a subset of vertices  $U \subseteq V$  is *k*-vertex separator of a graph  $G$ , if after removing  $U$  and incident edges, the remaining graph forms a union of connected components, each of size at most  $k$ .

Furthermore, let

$$S_G(k) = \min (|U| : U \text{ is a } k\text{-vertex separator of } G)$$

denote the size of the minimal *k*-vertex separator of graph  $G$ .

#### 2.1.2 Small Set Expansion Hypothesis

In this section we define the Small Set Expansion (SSE) Hypothesis introduced in [19]. Let  $G = (V, E)$  be an undirected  $d$ -regular graph.

► **Definition 8** (*Normalized edge expansion*). For a set  $S \subseteq V$  of vertices, denote  $\Phi_G(S)$  as the normalized edge expansion of  $S$ ,

$$\Phi_G(S) = \frac{|E(S, V \setminus S)|}{d|S|},$$

where  $|E(S, V \setminus S)|$  is the number of edges between  $S$  and  $V \setminus S$ .

The *Small Set Expansion Problem* with parameters  $\eta$  and  $\delta$ , denoted  $\text{SSE}(\eta, \delta)$ , asks whether  $G$  has a small set  $S$  which does not expand or all small sets of  $G$  are highly expanding.

► **Definition 9** ( $SSE(\eta, \delta)$ ). Given a regular graph  $G = (V, E)$ , distinguish between the following two cases:

- **Yes** There is a set of vertices  $S \subseteq V$  with  $|S| = \delta|V|$  and  $\Phi_G(S) \leq \eta$
- **No** For every set of vertices  $S \subseteq V$  with  $|S| = \delta|V|$  it holds that  $\Phi_G(S) \geq 1 - \eta$

The *Small Set Expansion Hypothesis* is the conjecture that deciding  $SSE(\eta, \delta)$  is NP-hard.

► **Conjecture 10** (Small Set Expansion Hypothesis [19]). *For every  $\eta > 0$ , there is a  $\delta > 0$  such that  $SSE(\eta, \delta)$  is NP-hard.*

We say that a problem is *SSE-hard* if it is at least as hard to solve as the SSE problem. The form of conjecture most relevant to our proof is the following “stronger” form of the SSE Hypothesis. [20] showed that the SSE-problem can be reduced to a quantitatively stronger form of itself. In order to state this version, we first need to define the *Gaussian noise stability*.

► **Definition 11** (Gaussian Noise Stability). Let  $\rho \in [-1, 1]$ . Define  $\Gamma_\rho : [0, 1] \mapsto [0, 1]$  by

$$\Gamma_\rho(\mu) = Pr[X \leq \Phi^{-1}(\mu) \wedge Y \leq \Phi^{-1}(\mu)]$$

where  $X$  and  $Y$  are jointly normal random variables with mean 0 and covariance matrix  $\begin{pmatrix} 1 & \rho \\ \rho & 1 \end{pmatrix}$ .

The only fact that we will use for stating the stronger form of SSEH is the asymptotic behavior of  $\Gamma_\rho(\mu)$  when  $\rho$  is close to 1 and  $\mu$  bounded away from 0.

► **Fact 12.** *There is a constant  $c > 0$  such that for all sufficiently small  $\epsilon$  and all  $\mu \in [1/10, 1/2]$ ,*<sup>5</sup>

$$\Gamma_{1-\epsilon}(\mu) \leq \mu(1 - c\sqrt{\epsilon}).$$

► **Conjecture 13** (SSE Hypothesis, Equivalent Formulation [20]). *For every integer  $q > 0$  and  $\epsilon, \gamma > 0$ , it is NP-hard to distinguish between the following two cases for a given regular graph  $G = (V, E)$ :*

- **Yes** There is a partition of  $V$  into  $q$  equi-sized sets  $S_1, \dots, S_q$  such that  $\Phi_G(S_i) \leq 2\epsilon$  for every  $1 \leq i \leq q$ .
- **No** For every  $S \subseteq V$ , letting  $\mu = |S|/|V|$ , it holds that  $\Phi_G(S) \geq 1 - (\Gamma_{1-\epsilon/2}(\mu) + \gamma)/\mu$ , where the  $\Gamma_{1-\epsilon/2}(\mu)$  is the Gaussian noise stability.

We present two remarks about the Conjecture 13 from [2], which are relevant to our proof of Theorem 2.

► **Remark 14.** [2] The **Yes** instance of Conjecture 13 implies that the number of edges leaving each  $S_i$  is at most  $4\epsilon|E|/q$ , so the total number of edges not contained in one of the  $S_i$  is at most  $2\epsilon|E|$ .

► **Remark 15.** [2] The **No** instance of Conjecture 13 implies that for  $\epsilon$  sufficiently small, there exists some constant  $c'$  such that  $\Phi_G(S) \geq c'\sqrt{\epsilon}$ , provided that  $\mu \in [1/10, 1/2]$  and setting  $\gamma \leq \sqrt{\epsilon}$ . In particular,  $|E(S, V \setminus S)| \geq \Omega(\sqrt{\epsilon}|E|)$ , for any  $|V|/10 \leq |S| \leq 9|V|/10$ .<sup>6</sup>

<sup>5</sup> Note that the lower bound on  $\mu$  can be taken arbitrarily close to 0. So the statement holds with  $\mu \in [c', 1/2]$  for any constant  $c' > 0$ .

<sup>6</sup> Recall that Fact 12 is true for  $\mu \in [c', 1/2]$  for any constant  $c' > 0$ . Therefore, Remark (15) can be strengthened and states, for any  $c'|V| \leq |S| \leq (1 - c')|V|$ ,  $|E(S, V \setminus S)| \geq \Omega(\sqrt{\epsilon}|E|)$ . This will be a useful fact for proving hardness of approximation of  $m(G, g)$  for finding many truthful nodes in Section 5 in full length paper [8].

Remark 14 follows from the definition of normalized edge expansion and the fact that sum of degree is two times number of edges. Remark 15 follows from Fact 12. The strong form of SSE Hypothesis 13, Remark 14 and Remark 15 will be particularly helpful for proving our SSE-hardness of approximation result (Theorem 2).

## 2.2 Preliminaries for Corruption Detection on Networks

We model networks as directed or undirected graphs, where each vertex in the network can be one of two types: truthful or corrupted. At times, we will informally call truthful vertices “good” and corrupt vertices “bad.” We say that the corrupt party has *budget*  $b$  if it can afford to corrupt at most  $b$  nodes of the graph. Given a vertex set  $V$ , and a budget  $b$ , the corrupt entity will choose to control a subset of nodes  $B \subseteq V$  under the constraint  $|B| \leq b$ . The rest of the graph remains as truthful vertices, i.e.,  $T = V \setminus B \subseteq V$ . We assume that there are more truthful than corrupt nodes ( $b < |V|/2$ ). It is easy to see that in the case where  $|B| \geq |T|$ , the corrupt nodes can prevent the identification of even one truthful node, by simulating truthful nodes (see e.g. [1]).

Each node audits and reports its (outgoing) neighbors’ identities. That is, each vertex  $u \in V$  will report the type of each  $v \in \mathcal{N}(u)$ , which is a vector in  $\{0, 1\}^{|\mathcal{N}(u)|}$ . Truthful nodes always report the truth, i.e., it reports its neighbor  $v \in T$  if  $v$  is truthful,  $v \in B$  if  $v$  is corrupt. The corrupt nodes report their neighbors’ identities adversarially. In summary, a strategy of the bad agents is composed of a strategy to take over at most  $b$  nodes on the graph, and reports on the nodes that neighbor them.

► **Definition 16** (Strategy for a corrupt party). A strategy for the corrupt party is a function that maps a graph  $G$  and budget  $b$  to a subset of nodes  $B$  with size  $|B| \leq b$ , and a set of reports that each node  $v \in B$  gives about its neighboring nodes,  $\mathcal{N}(v)$ .

► **Definition 17** (Computationally bounded corrupt party). We say that the corrupt party is computationally bounded if its strategy can only be a polynomial-time computable function.

The task for the central agency is to find a good node on this corrupted network, based on the reports. It is clear that the more budget the corrupt party has, the harder the task of finding one truthful node becomes. It was observed in [1] that, for any graph, it is not possible to find one good node if  $b \geq |V|/2$ . If  $b = 0$ , it is clear that the entire set  $V$  is truthful. Therefore, given an arbitrary graph  $G$ , there exists a critical number  $m(G)$ , such that if the bad party has budget lower than  $m(G)$ , it is always possible to find a good node; if the bad party has budget greater than or equal to  $m(G)$ , it may not be possible to find a good node. In light of this, we define the critical number of bad nodes on a graph  $G$ . First, we formally define what we mean when we say it is impossible to find a truthful node on a graph  $G$ .

► **Definition 18** (Impossibility of finding one truthful node). Given a graph  $G = (V, E)$ , the bad party’s budget  $b$  and reports, we say that it is *impossible to identify one truthful node* if for any  $v \in V$ , there exists a configuration of the identities of the nodes where  $v$  is bad, at most  $b$  nodes are bad, and the configuration is consistent with the given reports.

► **Definition 19** (Critical number of bad nodes on a graph  $G$ ,  $m(G)$ ). Given an arbitrary graph  $G = (V, E)$ , we define  $m(G)$  as the minimum number  $b$  such that there is a way to distribute  $b$  corrupt nodes and set their corresponding reports such that it is impossible to find *one* truthful node on the graph  $G$ , given  $G$ , the reports and that the bad party’s budget is at most  $b$ .



For example, for a star graph  $G$  with  $|V| \geq 5$ , the critical number of bad nodes is  $m(G) = 2$ . If there is at most 1 corrupt node on  $G$ , the central agency can always find a good node, thus  $m(G) > 1$ . If there are at most 2 bad nodes on  $G$ , then the bad party can control the center node and one of the leaves. Then for any node  $v$  in the graph, there exists a configuration where  $v$  is bad, only two nodes are assigned bad, and the configuration is consistent with observed reports. Therefore, it is impossible for central agency to find one good node,  $m(G) = 2$ .

Given a graph  $G$ , by definition there exists a set of nodes of size  $m(G)$  that can make it impossible to find a good node if they are corrupted. However, this does not mean that the corrupt party can necessarily find this set in polynomial time. Indeed, Theorem 2 establishes that they cannot always find this set in polynomial time if we assume the SSE Hypothesis (Conjecture 13) and that  $P \neq NP$ .

### 3 Proofs and Main Lemmas

In the following section, we state our main results by first presenting the close relation of our problem to the  $k$ -vertex separator problem. Then we use this characterization to prove Theorem 1. This characterization will additionally be useful for the proofs of Theorems 2 and 3, which we will sketch in Section 3.2, and provide in Section 3.3 in our full length paper [8].

#### 3.1 2-Approximation by Vertex Separation

► **Lemma 20** (2-Approximation by Vertex Separation). *The critical number of corrupt nodes for graph  $G$ ,  $m(G)$ , can be bounded by the minimal sum of  $k$ -vertex separator and  $k$ ,  $\min_k (S_G(k) + k)$ , up to a factor of 2. i.e.,*

$$\frac{1}{2} \min_k (S_G(k) + k) \leq m(G) \leq \min_k (S_G(k) + k)$$

**Proof of Lemma 20.** The direction  $m(G) \leq \min_k S_G(k) + k$  follows simply. Let  $k^* = \arg \min_k (S_G(k) + k)$ . If the corrupt party is given  $S_G(k^*) + k^*$  nodes to corrupt on the graph, it can first assign  $S_G(k^*)$  nodes to the separator, thus the remaining nodes are partitioned into components of size at most  $k^*$ . Then it arbitrarily assigns one of the components to be all bad nodes. The bad nodes in the connected components report the nodes in the same component as good, and report any node in the separator as bad. The nodes in the separator can effectively report however they want (e.g. report all neighboring nodes as bad). It is impossible to identify even one single good node, because all connected components of size  $k$  can potentially be bad, and all vertices in the separator are bad.

The direction  $1/2 \min_k (S_G(k) + k) \leq m(G)$  can be proved as follows. When there are  $b = m(G)$  corrupt nodes distributed optimally in  $G$ , it is impossible to find a single good node by definition, and therefore, in particular, the following algorithm (Algorithm 1) cannot always find a good node:

---

**Algorithm 1** Finding one truthful vertex on undirected graph  $G$ .

---

Input: Undirected graph  $G$

- If the reports on edge  $(u, v)$  does not equal to  $(u \in T, v \in T)$ , remove both  $u, v$  and any incident edges. Remove a pair of nodes in each round, until there are no bad reports left.
  - Call the remaining graph  $H$ . Declare the largest component of  $H$  as good.
-

Run Algorithm 1 on  $G$ , and suppose the first step terminates in  $i$  rounds, then:

- No remaining node reports neighbors as corrupt
- $|V| - 2i$  nodes remain in graph
- $\leq b - i$  bad nodes remain in the graph, because each time we remove an edge with bad report, and one of the end points must be a corrupt vertex.

Note that if two nodes report each other as good, they must be the same type (either both truthful, or both corrupt.) Since graph  $H$  only contains good reports, nodes within a connected component of  $H$  have the same types. If there exists a component of size larger than  $b - i$ , it exceeds bad party's budget, and must be all good. Therefore, Algorithm 1 would successfully find a good node.

Since Algorithm 1 cannot find a good node, the bad party must have the budget to corrupt the largest component of  $H$ , which means it has size at most  $b - i$ . Hence,  $S_G(b - i) \leq 2i$ . Plugging in  $b = m(G)$ , we get that

$$m(G) = \frac{2i}{2} + b - i \geq \min_k (S_G(k)/2 + k) \geq \frac{1}{2} \min_k (S_G(k) + k),$$

where the first inequality comes from  $2i \geq S_G(b - i)$ . ◀

Furthermore, the upper bound in Lemma 20 additionally tells us that if corrupt party's budget  $b \leq m(G)/2$ , the set output by Algorithm 1 is guaranteed to be good.

► **Theorem 1 (restated).** Fix a graph  $G$  and suppose that the corrupt party has a budget  $b \leq m(G)/2$ . Then the central agency can identify a truthful node, regardless of the strategy of the corrupt party, and without knowledge of either  $m(G)$  or  $b$ . Furthermore, the central agency's algorithm runs in linear time (in the number of edges in the graph  $G$ ).

**Proof of Theorem 1.** Suppose the corrupt party has budget  $b \leq m(G)/2$ . Run Algorithm 1. We remove  $2i$  nodes in the first step, and separate the remaining graph  $H$  into connected components. Notice each time we remove an edge with bad report, at least one of the end point is a corrupt vertex. So we have removed at most  $2b \leq m(G) \leq \lceil |V|/2 \rceil$  nodes. Therefore, the graph  $H$  is nonempty, and the nodes in any connected component of  $H$  have the same identity. Let  $k^* \geq 1$  be the size of the maximum connected component of  $H$ . We can conclude that  $S_G(k^*) \leq 2i$ , since  $2i$  is a possible size of  $k^*$ -vertex separator of  $G$ .

Notice there are at most  $b - i \leq m(G)/2 - i$  bad nodes in  $H$  by the same fact that at least one bad node is removed each round. By the upper bound in Lemma 20,

$$b - i \leq m(G)/2 - i \leq \min_k (S_G(k) + k)/2 - i \leq (2i + k^*)/2 - i \leq \frac{k^*}{2}.$$

Since  $k^* \geq 1$ , the connected component of size  $k^*$  exceeds the bad party's remaining budget  $k^*/2$ , and must be all good.

Algorithm 1 is linear time because it loops over all edges and removes any "bad" edge that does not have reports  $(T, T)$  (takes  $\leq |E|$  time when we use a list with "bad" edges at the front), and counts the size of the remaining components ( $\leq |V|$  time), and thus is linear in  $|E|$ . ◀

► **Remark 21.** Both bounds in Lemma 20 are tight. For the lower bound, consider a complete graph with an even number of nodes. For the upper bound, consider a complete bipartite graph with one side smaller than the other.

A tight lower bound and a tight upper bound example can be found in full-length version of our paper [8].

We end by discussing that the efficient algorithm given in this section does not address the regime when the budget of the bad party,  $b$ , falls in  $m(G)/2 < b \leq m(G)$ . Though by definition of  $m(G)$ , the central agency can find at least one truthful node as long as  $b \leq m(G)$ , by, for example, enumerating all possible assignments of good/bad nodes consistent with the report, and check the intersection of the assignment of good nodes. However, it is not clear that the central agency has a polynomial time algorithm for doing this. Of course, one can always run Algorithm 1, check whether the output set exceeds  $b - i/2$ , and concludes that the output set is truthful if that is the case. However, there is no guarantee that the output set will be larger than  $b - i/2$  if  $m(G)/2 < b \leq m(G)$ . We propose the following conjecture:

► **Conjecture 22.** *Fix a graph  $G$  and suppose that the corrupt party has a budget  $b$  such that  $m(G)/2 < b \leq m(G)$ . The problem of finding one truthful node given the graph  $G$ , bad party's budget  $b$  and the reports is NP-hard.*

### 3.2 SSE-Hardness of Approximation for $m(G)$

In this section, we present the hardness of approximation result for  $m(G)$  within any constant factor under the Small Set Expansion (SSE) Hypothesis [19]. Specifically, we give essential steps for proving Theorem 2.

► **Theorem 2 (restated).** For every  $\beta > 1$ , there is a constant  $\epsilon > 0$  such that the following is true. Given a graph  $G = (V, E)$ , it is SSE-hard to distinguish between the case where  $m(G) \leq \epsilon \cdot |V|$  and  $m(G) \geq \beta \cdot \epsilon \cdot |V|$ . Or in other words, the problem of approximating the critical number of corrupt nodes for a graph to within any constant factor is SSE-hard.

In order to prove Theorem 2, we construct a reduction similar to [2], and show that the bad party can control auxiliary graph of the **Yes** case of SSE with  $b = O(\epsilon|V'|)$  and cannot control the auxiliary graph of the **No** case of SSE with  $b = \Omega(\epsilon^{0.51}|V'|)$ . In the following, we present the arguments for the **No** case.

Given an undirected  $d$ -regular graph  $G = (V, E)$ , construct an auxiliary undirected graph  $G' = (V', E')$  in the following way [2]. Let  $r = d/2$ . For each vertex  $v^i \in V$ , make  $r$  copies of  $v^i$  and add to the vertex set of  $G'$ , denoted  $v_1^i, \dots, v_r^i$ . Denote the resulting set of vertices as  $\tilde{V} = V \times \{1, \dots, r\}$ . Each edge  $e^k \in E$  of  $G$  becomes a vertex in  $G'$ , denoted  $e^k$ . Denote this set of vertices as  $\tilde{E}$ . In other words,  $V' = \tilde{V} \cup \tilde{E} = V \times \{1, \dots, r\} \cup E$ . There exists an edge between a vertex  $v_j^i$  and a vertex  $e^k$  of  $G'$  if  $v^i$  and  $e^k$  were adjacent edge and vertex pair in  $G$ . Note that  $G'$  is a bipartite  $d$ -regular graph with  $d/2|V| + |E| = 2|E|$  vertices.

► **Lemma 23.** *Suppose  $q = 1/\epsilon$ , and  $G$  can be partitioned into  $q$  equi-sized sets  $S_1, \dots, S_q$  such that  $\Phi_G(S_i) \leq 2\epsilon$  for every  $1 \leq i \leq q$ . Then the bad party can control the auxiliary graph  $G'$  with at most  $3\epsilon|E| = 1.5\epsilon|V'|$  nodes.*

**Proof of Lemma 23.** Notice by Remark (2.1.1), the total number of edges in  $G$  not contained in one of the  $S_i$  is at most  $2\epsilon|E|$ .

This implies that a strategy for the bad party to control graph  $G'$  is as follows. Control vertex  $e^k \in \tilde{E}$  if  $e^k \in E$  is not contained in any of the  $S_i$ s in  $G$ . Call the set of such vertices  $E^* \subseteq \tilde{E}$ . Let  $S_i^* \subseteq V'$  be the set that contains all  $r$  copies of nodes in  $S_i \subseteq V$ . Control one of the  $S_i^*$ s, say  $S_1^*$ . Corrupt nodes in  $S_1^*$  report their neighbors in  $S_1^*$  as good, and report  $E^*$  as bad. Nodes in  $E^*$  can effectively report however they want; suppose they report every neighboring node as bad. Then, it is impossible to identify even one truthful node, since assigning any  $S_i^*$  as corrupt is consistent with the report and within bad party's budget.

If  $q = 1/\epsilon$ , this strategy amounts to controlling  $2\epsilon|E| + d/2 \cdot |V|/q = 3\epsilon|E|$  nodes on  $G'$ . Notice, this number is guaranteed to be smaller than  $1/2|V'|$ , as long as  $q > 4$ , because the bad party controls less than  $2/q$  of all the “edge vertices”  $\tilde{E}$ , and controls less than  $1/q$  of all the “vertex vertices”  $\tilde{V}$ . ◀

Note that, different from the argument in [2], we cannot take  $r$  to be arbitrarily large (e.g.  $> O(|V||E|)$ ). This is because when  $r$  is large,  $2\epsilon|E| + r \cdot |V|/q = O(\epsilon(|E| + |V'|)) = O(\epsilon|V'|)$ , and will not be comparable with the  $O(\sqrt{\epsilon}|E|)$  in Lemma 24.

► **Lemma 24.** *Let  $G = (V, E)$  be an undirected  $d$ -regular graph with the property that for every  $|V|/10 \leq |S| \leq 9|V|/10$  we have  $|E(S, V \setminus S)| \geq \Omega(\sqrt{\epsilon}|E|)$ . If bad party controls  $O(\epsilon^{0.51}|E|) = O(\epsilon^{0.51}|V'|) < 1/2|V'|$  nodes on the auxiliary graph  $G'$  constructed from  $G$ , we can always find a truthful node on  $G'$ .*

Combining Lemma 23 and Lemma 24, Theorem 2 follows in standard fashion. The proofs for Lemma 24 and Theorem 2 are presented in our full-length paper [8].

We also obtain the following Corollary 25 from Theorem 2.

► **Corollary 25.** *Assume the SSE Hypothesis and that  $P \neq NP$ . Fix any  $\beta > 1$ . There does not exist a polynomial-time algorithm that takes as input an arbitrary graph  $G = (V, E)$  and outputs a set of nodes  $S$  with size  $|S| \leq O(\beta \cdot m(G))$ , such that corrupting  $S$  prevents the central agency from finding a truthful node.*

In summary, the analysis in this section tells us that given an arbitrary graph, it is hard for bad party to corrupt the graph with minimal resources. Moreover, it is still hard for the bad party to corrupt the graph even if they are given a budget of  $\beta m(G)$ , for any  $\beta \geq 1$ . On the other hand, if the budget of the bad party is a factor of two less than  $m(G)$ , a good node can always be detected with an efficient algorithm, e.g. using Algorithm 1. This contrast highlights that the corruption detection problem of finding one good node (and, as later proven, finding any arbitrary fraction of good nodes) is easier for the good party and harder for the bad party.

Finally, the existence of efficient algorithms for good party to detect one good node on directed graphs (i.e. Theorem 4), and detect any fraction of the good nodes (i.e. Theorem 5) follows from analogous notions of vertex-separators to Definition 7. The hardness of approximation results for finding any arbitrary fraction of good nodes, i.e. 6 can be proven following similar constructions and arguments as in the proof sketch of Theorem 2. Then, we give a simple a gadget reduction that extends the result to  $\delta \in [1/2, 1)$ . The full details of this proof can be found in the full version of our paper, together with the proofs of Theorem 3, 4, 5, 6.

---

## References

- 1 Noga Alon, Elchanan Mossel, and Robin Pemantle. Corruption Detection on Networks. *CoRR*, abs/1505.05637, 2015. [arXiv:1505.05637](https://arxiv.org/abs/1505.05637).
- 2 Per Austrin, Toniann Pitassi, and Yu Wu. Inapproximability of Treewidth, One-Shot Pebbling, and Related Layout Problems. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques - 15th International Workshop, APPROX 2012, and 16th International Workshop, RANDOM 2012, Cambridge, MA, USA, August 15-17, 2012. Proceedings*, pages 13–24, 2012. doi:10.1007/978-3-642-32512-0\_2.
- 3 Walid Ben-Ameur, Mohamed-Ahmed Mohamed-Sidi, and José Neto. The  $k$ -separator problem: polyhedra, complexity and approximation results. *J. Comb. Optim.*, 29(1):276–307, 2015. doi:10.1007/s10878-014-9753-x.

- 4 Hans L. Bodlaender, John R. Gilbert, Hjálmtyr Hafsteinsson, and Ton Kloks. Approximating Treewidth, Pathwidth, Frontsize, and Shortest Elimination Tree. *J. Algorithms*, 18(2):238–255, 1995.
- 5 Anton T. Dahbura and Gerald M. Masson. An  $O(n^{2.5})$  Fault Identification Algorithm for Diagnosable Systems. *IEEE Trans. Computers*, 33(6):486–492, 1984. doi:10.1109/TC.1984.1676472.
- 6 Odd-Helge Fjeldstad. Fighting fiscal corruption: lessons from the Tanzania Revenue Authority. *Public Administration and Development: The International Journal of Management Research and Practice*, 23(2):165–175, 2003.
- 7 S. Louis Hakimi and A. T. Amin. Characterization of Connection Assignment of Diagnosable Systems. *IEEE Trans. Computers*, 23(1):86–88, 1974. doi:10.1109/T-C.1974.223782.
- 8 Yan Jin, Elchanan Mossel, and Govind Ramnarayan. Being Corrupt Requires Being Clever, But Detecting Corruption Doesn't. *arXiv*, 2018. arXiv:1809.10325.
- 9 Tiko Kameda, S Toida, and FJ Allan. A diagnosing algorithm for networks. *Information and Control*, 29(2):141–148, 1975.
- 10 David Kempe, Jon M. Kleinberg, and Éva Tardos. Maximizing the Spread of Influence through a Social Network. *Theory of Computing*, 11:105–147, 2015. doi:10.4086/toc.2015.v011a004.
- 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. doi:10.1109/CCC.2002.1004334.
- 12 Jon G Kuhl and Sudhakar M Reddy. Distributed fault-tolerance for large multiprocessor systems. In *Proceedings of the 7th annual symposium on Computer Architecture*, pages 23–30. ACM, 1980.
- 13 Euiwoong Lee. Partitioning a Graph into Small Pieces with Applications to Path Transversal. In *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2017, Barcelona, Spain, Hotel Porta Fira, January 16-19*, pages 1546–1558, 2017. doi:10.1137/1.9781611974782.101.
- 14 Shachindra N. Maheshwari and S. Louis Hakimi. On Models for Diagnosable Systems and Probabilistic Fault Diagnosis. *IEEE Trans. Computers*, 25(3):228–236, 1976.
- 15 Thebeth Rufaro Mukwembi and Simon Mukwembi. Corruption and its detection: a graph-theoretic approach. *Computational and Mathematical Organization Theory*, 23(2):293–300, June 2017. doi:10.1007/s10588-016-9227-z.
- 16 Richard P Nielsen. Corruption networks and implications for ethical corruption reform. *Journal of Business ethics*, 42(2):125–149, 2003.
- 17 Maarten Oosten, Jeroen HGC Rutten, and Frits CR Spieksma. Disconnecting graphs by removing vertices: a polyhedral approach. *Statistica Neerlandica*, 61(1):35–60, 2007.
- 18 Franco P. Preparata, Gernot Metze, and Robert T. Chien. On the Connection Assignment Problem of Diagnosable Systems. *IEEE Trans. Electronic Computers*, 16(6):848–854, 1967. doi:10.1109/PGEC.1967.264748.
- 19 Prasad Raghavendra and David Steurer. Graph expansion and the unique games conjecture. In *Proceedings of the 42nd ACM Symposium on Theory of Computing, STOC 2010, Cambridge, Massachusetts, USA, 5-8 June 2010*, pages 755–764, 2010. doi:10.1145/1806689.1806788.
- 20 Prasad Raghavendra, David Steurer, and Madhur Tulsiani. Reductions between Expansion Problems. In *Proceedings of the 27th Conference on Computational Complexity, CCC 2012, Porto, Portugal, June 26-29, 2012*, pages 64–73, 2012. doi:10.1109/CCC.2012.43.
- 21 Gregory F. Sullivan. A Polynomial Time Algorithm for Fault Diagnosability. In *FOCS*, pages 148–156. IEEE Computer Society, 1984.
- 22 Jie Xu and Shi-ze Huang. Sequentially t-Diagnosable Systems: A Characterization and Its Applications. *IEEE Trans. Computers*, 44(2):340–345, 1995. doi:10.1109/12.364544.

# Simulating Random Walks on Graphs in the Streaming Model

Ce Jin

Institute for Interdisciplinary Information Sciences, Tsinghua University, Beijing, China  
jinc16@mails.tsinghua.edu.cn

---

## Abstract

We study the problem of approximately simulating a  $t$ -step random walk on a graph where the input edges come from a single-pass stream. The straightforward algorithm using reservoir sampling needs  $O(nt)$  words of memory. We show that this space complexity is near-optimal for directed graphs. For undirected graphs, we prove an  $\Omega(n\sqrt{t})$ -bit space lower bound, and give a near-optimal algorithm using  $O(n\sqrt{t})$  words of space with  $2^{-\Omega(\sqrt{t})}$  simulation error (defined as the  $\ell_1$ -distance between the output distribution of the simulation algorithm and the distribution of perfect random walks). We also discuss extending the algorithms to the turnstile model, where both insertion and deletion of edges can appear in the input stream.

**2012 ACM Subject Classification** Theory of computation → Streaming models

**Keywords and phrases** streaming models, random walks, sampling

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2019.46

**Acknowledgements** I would like to thank Professor Jelani Nelson for introducing this problem to me, advising this project, and giving many helpful comments on my writeup.

## 1 Introduction

Graphs of massive size are used for modeling complex systems that emerge in many different fields of study. Challenges arise when computing with massive graphs under memory constraints. In recent years, graph streaming has become an important model for computation on massive graphs. Many space-efficient streaming algorithms have been designed for solving classical graph problems, including connectivity [2], bipartiteness [2], minimum spanning tree [2], matching [8, 12, 1], spectral sparsifiers [14, 13], etc. We will define the streaming model in Section 1.1.

Random walks on graphs are stochastic processes that have many applications, such as connectivity testing [17], clustering [18, 3, 4, 5], sampling [11] and approximate counting [10]. Since random walks are a powerful tool in algorithm design, it is interesting to study them in the streaming setting. A natural problem is to find the space complexity of simulating random walks in graph streams. Das Sarma et al. [7] gave a multi-pass streaming algorithm that simulates a  $t$ -step random walk on a directed graph using  $O(\sqrt{t})$  passes and only  $O(n)$  space. By further extending this algorithm and combining with other ideas, they obtained space-efficient algorithms for estimating PageRank on graph streams. However, their techniques crucially rely on reading multiple passes of the input stream.

In this paper, we study the problem of simulating random walks in the *one-pass* streaming model. We show space lower bounds for both directed and undirected versions of the problem, and present algorithms that nearly match with the lower bounds. We summarize our results in Section 1.3.



© Ce Jin;

licensed under Creative Commons License CC-BY

10th Innovations in Theoretical Computer Science (ITCS 2019).

Editor: Avrim Blum; Article No. 46; pp. 46:1–46:15

Leibniz International Proceedings in Informatics



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



### 1.1 One-pass streaming model

Let  $G = (V, E)$  be a graph with  $n$  vertices. In the insertion-only model, the input graph  $G$  is defined by a stream of edges  $(e_1, \dots, e_m)$  seen in arbitrary order, where each edge  $e_i$  is specified by its two endpoints  $u_i, v_i \in V$ . An algorithm must process the edges of  $G$  in the order that they appear in the input stream. The edges can be directed or undirected, depending on the problem setting. Sometimes we allow multiple edges in the graph, where the multiplicity of an edge equals its number of occurrences in the input stream.

In the turnstile model, we allow both insertion and deletion of edges. The input is a stream of updates  $((e_1, \Delta_1), (e_2, \Delta_2), \dots)$ , where  $e_i$  encodes an edge and  $\Delta_i \in \{1, -1\}$ . The multiplicity of edge  $e$  is  $f(e) = \sum_{e_i=e} \Delta_i$ . We assume  $f(e) \geq 0$  always holds for every edge  $e$ .

### 1.2 Random walks

Let  $f(u, v)$  denote the multiplicity of edge  $(u, v)$ . The degree of  $u$  is defined by  $d(u) = \sum_{v \in V} f(u, v)$ . A  $t$ -step random walk starting from a vertex  $s \in V$  is a random sequence of vertices  $v_0, v_1, \dots, v_t$  where  $v_0 = s$  and  $v_i$  is a vertex uniformly randomly chosen from the vertices that  $v_{i-1}$  connects to, i.e.,  $\mathbb{P}[v_i = v | v_{i-1} = u] = f(u, v)/d(u)$ . Let  $\mathcal{RW}_{s,t} : V^{t+1} \rightarrow [0, 1]$  denote the distribution of  $t$ -step random walks starting from  $s$ , defined by<sup>1</sup>

$$\mathcal{RW}_{s,t}(v_0, \dots, v_t) = \mathbf{1}[v_0 = s] \prod_{i=0}^{t-1} \frac{f(v_i, v_{i+1})}{d(v_i)}. \quad (1)$$

For two distributions  $P, Q$ , we denote by  $|P - Q|_1$  their  $\ell_1$  distance. We say that a randomized algorithm can simulate a  $t$ -step random walk starting from  $v_0$  within error  $\varepsilon$ , if the distribution  $\mathbb{P}_w$  of its output  $w \in V^{t+1}$  satisfies  $|\mathbb{P}_w - \mathcal{RW}_{v_0,t}|_1 \leq \varepsilon$ . We say the random walk simulation is perfect if  $\varepsilon = 0$ .

We study the problem of simulating a  $t$ -step random walk within error  $\varepsilon$  in the streaming model using small space. We assume the length  $t$  is specified at the beginning. Then the algorithm reads the input stream. When a query with parameter  $v_0$  comes, the algorithm should simulate and output a  $t$ -step random walk starting from vertex  $v_0$ .

It is without loss of generality to assume that the input graph has no self-loops. If we can simulate a random walk on the graph with self-loops removed, we can then turn it into a random walk of the original graph by simply inserting self-loops after  $u$  with probability  $d_{\text{self}}(u)/d(u)$ . The values  $d_{\text{self}}(u), d(u)$  can be easily maintained by a streaming algorithm using  $O(n)$  words.

The random walk is not well-defined when it starts from a vertex  $u$  with  $d(u) = 0$ . For undirected graphs, this can only happen at the beginning of the random walk, and we simply let our algorithm return FAIL if  $d(v_0) = 0$ . For directed graphs, one way to fix this is to continue the random walk from  $v_0$ , by adding an edge  $(u, v_0)$  for every vertex  $u$  with  $d(u) = 0$ . We will not deal with  $d(u) = 0$  in the following discussion.

### 1.3 Our results

We will use  $\log x = \log_2 x$  throughout this paper.

The following two theorems give space lower bounds on directed and undirected versions of the problem. Note that the lower bounds hold even for simple graphs<sup>2</sup>.

<sup>1</sup> For a statement  $p$ , define  $\mathbf{1}[p] = 1$  if  $p$  is true, and  $\mathbf{1}[p] = 0$  if  $p$  is false.

<sup>2</sup> A *simple* graph is a graph with no multiple edges.



► **Theorem 1.** For  $t \leq n/2$ , simulating a  $t$ -step random walk on a simple directed graph in the insertion-only model within error  $\varepsilon = \frac{1}{3}$  requires  $\Omega(nt \log(n/t))$  bits of memory.

► **Theorem 2.** For  $t = O(n^2)$ , simulating a  $t$ -step random walk on a simple undirected graph in the insertion-only model within error  $\varepsilon = \frac{1}{3}$  requires  $\Omega(n\sqrt{t})$  bits of memory.

Theorem 3 and Theorem 4 give near optimal space upper bounds for the problem in the insertion-only streaming model.

► **Theorem 3.** We can simulate a  $t$ -step random walk on a directed graph in the insertion-only model perfectly using  $O(nt)$  words<sup>3</sup> of memory. For simple directed graphs, the memory can be reduced to  $O(nt \log(n/t))$  bits, assuming  $t \leq n/2$ .

► **Theorem 4.** We can simulate a  $t$ -step random walk on an undirected graph in the insertion-only model within error  $\varepsilon$  using  $O\left(n\sqrt{t} \cdot \frac{q}{\log q}\right)$  words of memory, where  $q = 2 + \frac{\log(1/\varepsilon)}{\sqrt{t}}$ . In particular, the algorithm uses  $O(n\sqrt{t})$  words of memory when  $\varepsilon = 2^{-\Theta(\sqrt{t})}$ .

Our algorithms also extend to the turnstile model.

► **Theorem 5.** We can simulate a  $t$ -step random walk on a directed graph in the turnstile model within error  $\varepsilon$  using  $O(n(t + \log \frac{1}{\varepsilon}) \log^2 \max\{n, 1/\varepsilon\})$  bits of memory.

► **Theorem 6.** We can simulate a  $t$ -step random walk on an undirected graph in the turnstile model within error  $\varepsilon$  using  $O(n(\sqrt{t} + \log \frac{1}{\varepsilon}) \log^2 \max\{n, 1/\varepsilon\})$  bits of memory.

## 2 Directed graphs in the insertion-only model

The simplest algorithm uses  $O(n^2)$  words of space (or only  $O(n^2)$  bits, if we assume the graph is simple) to store the adjacency matrix of the graph. When  $t \ll n$ , a better solution is to use reservoir sampling.

► **Lemma 7** (Reservoir sampling). Given a stream of  $n$  items as input, we can uniformly sample  $m$  of them without replacement using  $O(m)$  words of memory.

We can also sample  $m$  items from the stream *with* replacement in  $O(m)$  words of memory using  $m$  independent reservoir samplers each with capacity 1.

► **Theorem 8.** We can simulate a  $t$ -step random walk on a directed graph in the insertion-only model perfectly using  $O(nt)$  words of memory.

**Proof.** For each vertex  $u \in V$ , we sample  $t$  edges  $e_{u,1}, \dots, e_{u,t}$  outgoing from  $u$  with replacement. Then we perform a random walk using these edges. When  $u$  is visited for the  $i$ -th time ( $i \leq t$ ), we go along edge  $e_{u,i}$ . ◀

By treating an undirected edge as two opposite directed edges, we can achieve the same space complexity in undirected graphs.

Now we show a space lower bound for the problem. We will use a standard result from communication complexity.

► **Definition 9.** In the INDEX problem, Alice has an  $n$ -bit vector  $X \in \{0, 1\}^n$  and Bob has an index  $i \in [n]$ . Alice sends a message to Bob, and then Bob should output the bit  $X_i$ .

<sup>3</sup> A word has  $\Theta(\log \max\{n, m\})$  bits.

► **Lemma 10** ([15]). *For any constant  $1/2 < c \leq 1$ , solving the INDEX problem with success probability  $c$  requires sending  $\Omega(n)$  bits.*

► **Theorem 11.** *For  $t \leq n/2$ , simulating a  $t$ -step random walk on a simple directed graph in the insertion-only model within error  $\varepsilon = \frac{1}{3}$  requires  $\Omega(nt \log(n/t))$  bits of memory.*

**Proof.** We prove by showing a reduction from the INDEX problem. Before the protocol starts, Alice and Bob agree on a family  $\mathcal{F}$  of  $t$ -subsets of  $[n]$ <sup>4</sup> such that the condition  $|S \cap S'| < t/2$  is satisfied for every  $S, S' \in \mathcal{F}, S \neq S'$ . For two independent uniform random  $t$ -subsets  $S, S' \subseteq [n]$ , let  $p = \mathbb{P}[|S \cap S'| \geq t/2] \leq \binom{t}{t/2} (\frac{t}{n})^{t/2} < (\frac{4t}{n})^{t/2}$ . By union bound over all pairs of subsets, a randomly generated family  $\mathcal{F}$  satisfies the condition with probability at least  $1 - \binom{|\mathcal{F}|}{2} p$ , which is positive when  $|\mathcal{F}| = \lceil \sqrt{1/p} \rceil \geq (\frac{n}{4t})^{t/4}$ . So we can choose such family  $\mathcal{F}$  with  $\log |\mathcal{F}| = \Omega(t \log(n/t))$ .

Assume  $|\mathcal{F}|$  is a power of two. Alice encodes  $n \log |\mathcal{F}|$  bits as follows. Let  $G$  be a directed graph with vertex set  $\{v_0, v_1, \dots, v_{2n}\}$ . For each vertex  $u \in \{v_{n+1}, v_{n+2}, \dots, v_{2n}\}$ , Alice chooses a set  $S_u \in \mathcal{F}$ , and inserts an edge  $(u, v_i)$  for every  $i \in S_u$ .

Suppose Bob wants to query  $S_u$ . He adds an edge  $(v, u)$  for every  $v \in \{v_0, v_1, v_2, \dots, v_n\}$ , and then simulates a random walk starting from  $v_0$ . The random walk visits  $u$  every two steps, and it next visits  $v_i$  for some random  $i \in S_u$ . At least  $t/2$  different elements from  $S_u$  can be seen in  $2t$  samples with probability at least  $1 - \binom{t}{t/2} (\frac{1}{2})^{2t} \geq 1 - 2^{-t}$ , so  $S_u$  can be uniquely determined by an  $O(t)$ -step random walk (simulated within error  $\varepsilon$ ) with probability  $1 - 2^{-t} - \frac{\varepsilon}{2} > \frac{1}{2}$ . By Lemma 10, the space usage for simulating the  $O(t)$ -step random walk is at least  $\Omega(n \log |\mathcal{F}|) = \Omega(nt \log(n/t))$  bits. The theorem is proved by scaling down  $n$  and  $t$  by a constant factor. ◀

For simple graphs, we can achieve an upper bound of  $O(nt \log(n/t))$  bits.

► **Theorem 12.** *For  $t \leq n/2$ , we can simulate a  $t$ -step random walk on a simple directed graph in the insertion-only model perfectly using  $O(nt \log(n/t))$  bits of memory.*

**Proof.** For every  $u \in V$ , we run a reservoir sampler with capacity  $t$ , which samples (at most)  $t$  edges from  $u$ 's outgoing edges *without* replacement. After reading the entire input stream, we begin simulating the random walk. When  $u$  is visited during the simulation, in the next step we choose at random an outgoing edge used before with probability  $d_{\text{used}}(u)/d(u)$ , or an unused edge from the reservoir sampler with probability  $1 - d_{\text{used}}(u)/d(u)$ , where  $d_{\text{used}}(u)$  is the number of edges in  $u$ 's sampler that are previously used in the simulation. We maintain a  $t$ -bit vector to keep track of these used samples.

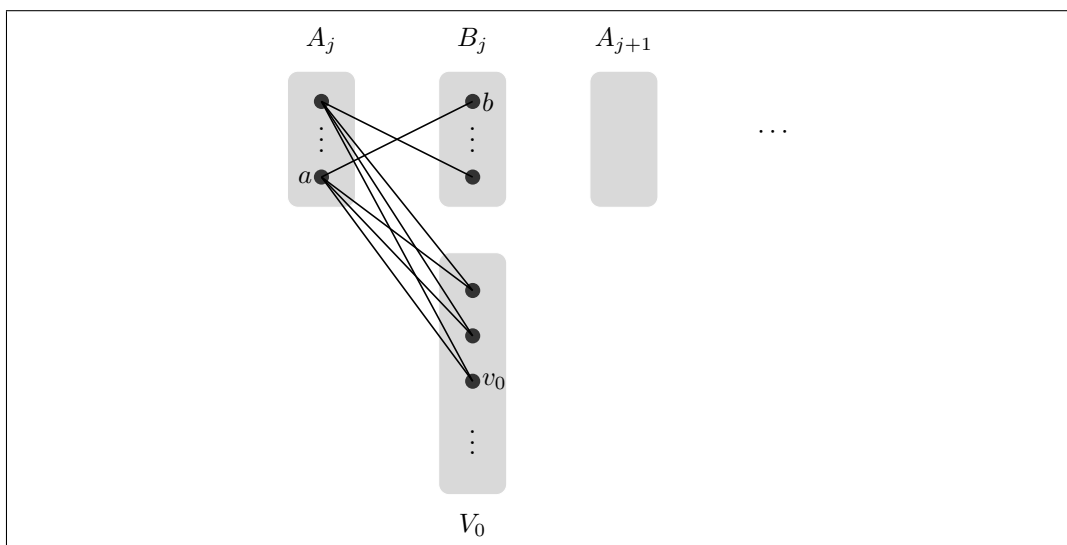
The number of different possible states of a sampler is at most  $\sum_{0 \leq i \leq t} \binom{n}{i} \leq (t+1) \binom{en}{t}$ , so it can be encoded using  $\lceil \log \left( (t+1) \binom{en}{t} \right) \rceil = O(t \log(n/t))$  bits. The total space is  $O(nt \log(n/t))$  bits. ◀

### 3 Undirected graphs in the insertion-only model

#### 3.1 A space lower bound

► **Theorem 13.** *For  $t = O(n^2)$ , simulating a  $t$ -step random walk on a simple undirected graph in the insertion-only model within error  $\varepsilon = \frac{1}{3}$  requires  $\Omega(n\sqrt{t})$  bits of memory.*

<sup>4</sup> Define  $[n] = \{1, 2, \dots, n\}$ . A  $t$ -subset is a subset of size  $t$ .



■ **Figure 1** Proof of Theorem 13.

**Proof.** Again we show a reduction from the INDEX problem.

Alice encodes  $\Omega(n\sqrt{t})$  bits as follows. Let  $G$  be an undirected graph with vertex set  $V_0 \cup V_1 \cup \dots \cup V_{n/\sqrt{t}}$ , where each  $V_j$  has size  $2\sqrt{t}$ , and the starting vertex  $v_0 \in V_0$ . For each  $j \geq 1$ ,  $V_j$  is divided into two subsets  $A_j, B_j$  with size  $\sqrt{t}$  each, and Alice encodes  $|A_j| \times |B_j| = t$  bits by inserting a subset of edges from  $\{(u, v) : u \in A_j, v \in B_j\}$ . In total she encodes  $t \cdot n/\sqrt{t} = n\sqrt{t}$  bits.

Suppose Bob wants to query some bit, i.e., he wants to see whether  $a$  and  $b$  are connected by an edge. Assume  $(a, b) \in A_j \times B_j$ . He adds an edge  $(u, v)$  for every  $u \in A_j$  and every  $v \in V_0$  (see Figure 1). A perfect random walk starting from  $v_0 \in V_0$  will be inside the bipartite subgraph  $(A_j, B_j \cup V_0)$ . Suppose the current vertex of the perfect random walk is  $v_i \in A_j$ . If  $a, b$  are connected by an edge, then

$$\begin{aligned}
 & \mathbb{P}[(v_{i+2}, v_{i+3}) = (a, b) \mid v_i] \\
 & \geq \mathbb{P}[v_{i+1} \in V_0 \mid v_i] \mathbb{P}[v_{i+2} = a \mid v_{i+1} \in V_0] \mathbb{P}[v_{i+3} = b \mid v_{i+2} = a] \\
 & \geq \frac{|V_0|}{|V_0| + |B_j|} \cdot \frac{1}{|A_j|} \cdot \frac{1}{|V_0| + |B_j|} \\
 & \geq \frac{2}{9t},
 \end{aligned}$$

so in every four steps the edge  $(a, b)$  is passed with probability  $\Omega(\frac{1}{t})$ . Then a  $O(t)$ -step perfect random walk will pass the edge  $(a, b)$  with probability 0.9. Hence Bob can know whether the edge  $(a, b)$  exists by looking at the random walk (simulated within error  $\varepsilon$ ) with success probability  $0.9 - \frac{\varepsilon}{2} > 1/2$ . By Lemma 10, the space usage for simulating the  $O(t)$ -step random walk is at least  $\Omega(n\sqrt{t})$  bits. The theorem is proved by scaling down  $n$  and  $t$  by a constant factor. ◀

### 3.2 An algorithm for simple graphs

Now we describe our algorithm for undirected graphs in the insertion-only model. As a warm-up, we consider simple graphs in this section. We will deal with multi-edges in Section 3.3.

### Intuition

We start by informally explaining the intuition of our algorithm for simple undirected graphs.

We maintain a subset of  $O(n\sqrt{t})$  edges from the input graph, and use them to simulate the random walk after reading the entire input stream.

For a vertex  $u$  with degree smaller than  $\sqrt{t}$ , we can afford to store all its neighboring edges in memory. For  $u$  with degree greater than  $\sqrt{t}$ , we can only sample and store  $O(\sqrt{t})$  of its neighboring edges. During the simulation, at every step we first toss a coin to decide whether the next vertex has small degree or large degree. In the latter case, we have to pick a sampled neighboring edge and walk along it. If all sampled neighboring edges have already been used, our algorithm fails. Using the large degree and the fact that edges are undirected, we can show that the failure probability is low.

### Description of the algorithm

We divide the vertices into two types according to their degrees: the set of *big* vertices  $B = \{u \in V : d(u) \geq C + 1\}$ , and the set of *small* vertices  $S = \{u \in V : d(u) \leq C\}$ , where parameter  $C$  is a positive integer to be determined later.

We use *arc*  $(u, v)$  to refer to an edge when we want to specify the direction  $u \rightarrow v$ . So an undirected edge  $(u, v)$  corresponds to two different<sup>5</sup> arcs,  $\text{arc}(u, v)$  and  $\text{arc}(v, u)$ .

We say an arc  $(u, v)$  is *important* if  $v \in S$ , or *unimportant* if  $v \in B$ . Denote the set of important arcs by  $E_1$ , and the set of unimportant arcs by  $E_0$ . The total number of important arcs equals  $\sum_{s \in S} d(s) \leq |S|C$ , so it is possible to store  $E_1$  in  $O(nC)$  words of space.

The set  $E_0$  of unimportant arcs can be huge, so we only store a subset of  $E_0$ . For every vertex  $u$ , we sample with replacement  $C$  unimportant arcs outgoing from  $u$ , denoted by  $a_{u,1}, \dots, a_{u,C}$ .

To maintain the set  $E_1$  of important arcs and the samples of unimportant arcs after every edge insertion, we need to handle the events when some small vertex becomes big. This procedure is straightforward, as described by `PROCESSINPUT` in Figure 2. Since  $|E_1|$  never exceeds  $nC$ , and each of the  $n$  samplers uses  $O(C)$  words of space, the overall space complexity is  $O(nC)$  words.

We begin simulating the random walk after `PROCESSINPUT` finishes. When the current vertex of the random walk is  $v$ , with probability  $d_1(v)/d(v)$  the next step will be along an important arc, where  $d_1(v)$  denotes the number of important arcs outgoing from  $v$ . In this case we simply choose a uniform random vertex from  $\{u : (v, u) \in E_1\}$  as the next vertex. However, if the next step is along an unimportant arc, we need to choose an unused sample  $a_{v,j}$  and go along this arc. If at this time all  $C$  samples  $a_{v,j}$  are already used, then our algorithm fails (and is allowed to return an arbitrary walk). The pseudocode of this simulating procedure is given in Figure 3.

In a walk  $w = (v_0, \dots, v_t)$ , we say vertex  $u$  *fails* if  $|\{i : v_i = u \text{ and } (v_i, v_{i+1}) \in E_0\}| > C$ . If no vertex fails in  $w$ , then our algorithm will successfully return  $w$  with probability  $\mathcal{RW}_{v_0,t}(w)$ . Otherwise our algorithm will fail after some vertex runs out of the sampled unimportant arcs. To ensure the output distribution is  $\varepsilon$ -close to  $\mathcal{RW}_{v_0,t}$  in  $\ell_1$  distance, it suffices to make our algorithm fail with probability at most  $\varepsilon/2$ , by choosing a large enough capacity  $C$ .

<sup>5</sup> We have assumed no self-loops exist, so  $u \neq v$ .

```

procedure INSERTARC( $u, v$ )
   $d(v) \leftarrow d(v) + 1$ 
  if  $d(v) = C + 1$  then                                     ▷  $v$  changes from small to big
    for  $x \in V$  such that  $(x, v) \in E_1$  do                 ▷ arc  $(x, v)$  becomes unimportant
       $E_1 \leftarrow E_1 \setminus \{(x, v)\}$ 
      Feed arc  $(x, v)$  into  $x$ 's sampler
    end for
  end if
  if  $d(v) \leq C$  then                                     ▷  $v \in S$ 
     $E_1 \leftarrow E_1 \cup \{(u, v)\}$ 
  else                                                       ▷  $v \in B$ 
    Feed arc  $(u, v)$  into  $u$ 's sampler
  end if
end procedure
procedure PROCESSINPUT
   $E_1 \leftarrow \emptyset$                                      ▷ Set of important arcs
  for  $u \in V$  do
     $d(u) \leftarrow 0$ 
    Initialize  $u$ 's sampler (initially empty) which maintains  $a_{u,1}, \dots, a_{u,C}$ 
  end for
  for undirected edge  $(u, v)$  in the input stream do
    INSERTARC( $u, v$ )
    INSERTARC( $v, u$ )
  end for
end procedure

```

■ **Figure 2** Pseudocode for processing the input stream (for simple undirected graphs).

```

procedure SIMULATERANDOMWALK( $v_0, t$ )
  for  $v \in V$  do
     $c(v) \leftarrow 0$                                        ▷ counter of used samples
  end for
  for  $i = 0, \dots, t - 1$  do
     $N_1 \leftarrow \{u : (v_i, u) \in E_1\}$ 
     $x \leftarrow$  uniformly random integer from  $\{1, 2, \dots, d(v_i)\}$ 
    if  $x \leq |N_1|$  then
       $v_{i+1} \leftarrow$  uniformly random vertex from  $N_1$ 
    else
       $j \leftarrow c(v_i) + 1$ 
       $c(v_i) \leftarrow j$ 
      if  $j > C$  then return FAIL
    else
       $v_{i+1} \leftarrow u$ , where  $(v_i, u) = a_{v_i, j}$ 
    end if
  end if
  end for
  return  $(v_0, \dots, v_t)$ 
end procedure

```

■ **Figure 3** Pseudocode for simulating a  $t$ -step random walk starting from  $v_0$ .

## 46:8 Simulating Random Walks on Graphs in the Streaming Model

To bound the probability  $\mathbb{P}[\text{at least one vertex fails} \mid v_0 = s]^6$ , we will bound the individual failure probability of every vertex, and then use union bound.

► **Lemma 14.** *Suppose for every  $u \in V$ ,  $\mathbb{P}[u \text{ fails} \mid v_0 = u] \leq \delta$ . Then for any starting vertex  $s \in V$ ,  $\mathbb{P}[\text{at least one vertex fails} \mid v_0 = s] \leq t\delta$ .*

**Proof.** Fix a starting vertex  $s$ . For any particular  $u \in V$ ,

$$\begin{aligned} & \mathbb{P}[u \text{ fails} \mid v_0 = s] \\ &= \mathbb{P}[u \text{ fails, and } \exists i \leq t-1, v_i = u \mid v_0 = s] \\ &= \mathbb{P}[\exists i \leq t-1, v_i = u \mid v_0 = s] \mathbb{P}[u \text{ fails} \mid v_0 = s, \text{ and } \exists i \leq t-1, v_i = u] \\ &\leq \mathbb{P}[\exists i \leq t-1, v_i = u \mid v_0 = s] \mathbb{P}[u \text{ fails} \mid v_0 = u] \\ &\leq \mathbb{P}[\exists i \leq t-1, v_i = u \mid v_0 = s] \cdot \delta. \end{aligned}$$

By union bound,

$$\begin{aligned} & \mathbb{P}[\text{at least one vertex fails} \mid v_0 = s] \\ &\leq \sum_{u \in V} \mathbb{P}[u \text{ fails} \mid v_0 = s] \\ &\leq \sum_{u \in V} \mathbb{P}[\exists i \leq t-1, v_i = u \mid v_0 = s] \cdot \delta \\ &= \mathbb{E}[\text{number of distinct vertices visited in } \{v_0, \dots, v_{t-1}\} \mid v_0 = s] \cdot \delta \\ &\leq t\delta. \end{aligned} \quad \blacktriangleleft$$

► **Lemma 15.** *We can choose integer parameter  $C = O\left(\sqrt{t} \cdot \frac{q}{\log q}\right)$ , where  $q = 2 + \frac{\log(1/\delta)}{\sqrt{t}}$ , so that  $\mathbb{P}[u \text{ fails} \mid v_0 = u] \leq \delta$  holds for every  $u \in V$ .*

**Proof.** Let  $d_0(u) = |\{v : (u, v) \in E_0\}|$ .

For any  $u \in V$ ,

$$\begin{aligned} & \mathbb{P}[u \text{ fails} \mid v_0 = u] \\ &\leq \mathbb{P}[u \text{ fails} \mid v_0 = u, (v_0, v_1) \in E_0]. \end{aligned}$$

We rewrite this probability as the sum of probabilities of possible random walks in which  $u$  fails. Recall that  $u$  fails if and only if  $|\{i : v_i = u, (v_i, v_{i+1}) \in E_0\}| \geq C + 1$ . In the summation over possible random walks, we only keep the shortest prefix  $(v_0, \dots, v_k)$  in which  $u$  fails, i.e., the last step  $(v_{k-1}, v_k)$  is the  $(C + 1)$ -st time walking along an unimportant arc outgoing from  $u$ . We have

$$\begin{aligned} & \mathbb{P}[u \text{ fails} \mid v_0 = u, (v_0, v_1) \in E_0] \\ &= \sum_{k \leq t} \sum_{\text{walk}(v_0, \dots, v_k)} \mathbf{1} \left[ \begin{array}{l} v_0 = v_{k-1} = u, (v_0, v_1), (v_{k-1}, v_k) \in E_0, \\ |\{i : v_i = u, (v_i, v_{i+1}) \in E_0\}| = C + 1 \end{array} \right] \frac{1}{d_0(u)} \prod_{i=1}^{k-1} \frac{1}{d(v_i)} \\ &= \sum_{k \leq t} \sum_{\text{walk}(v_0, \dots, v_{k-1})} \mathbf{1} \left[ \begin{array}{l} v_0 = v_{k-1} = u, (v_0, v_1) \in E_0, \\ |\{i : v_i = u, (v_i, v_{i+1}) \in E_0\}| = C \end{array} \right] \prod_{i=1}^{k-1} \frac{1}{d(v_i)}. \end{aligned} \quad (2)$$

<sup>6</sup> If not specified, assume the probability space is over all  $t$ -step random walks  $(v_0, \dots, v_t)$  starting from  $v_0$ .

Let  $v'_i = v_{k-1-i}$ . Since the graph is undirected, the vertex sequence  $(v'_0, \dots, v'_{k-1})$  (the reversal of walk  $(v_0, \dots, v_{k-1})$ ) is also a walk starting from and ending at  $u$ . So the summation (2) equals

$$\begin{aligned} & \sum_{k \leq t} \sum_{\text{walk}(v'_0, \dots, v'_{k-1})} \mathbf{1} \left[ v'_0 = v'_{k-1} = u, (v'_{k-1}, v'_{k-2}) \in E_0, \right. \\ & \quad \left. |\{i : v'_i = u, (v'_i, v'_{i-1}) \in E_0\}| \geq C \right] \prod_{i=0}^{k-2} \frac{1}{d(v'_i)} \\ &= \mathbb{P}_{\text{random walk}(v'_0, \dots, v'_{t-1})} \left[ |\{i : v'_i = u, (v'_i, v'_{i-1}) \in E_0\}| \geq C \mid v'_0 = u \right]. \end{aligned}$$

Recall that  $(v'_i, v'_{i-1}) \in E_0$  if and only if  $v'_{i-1} \in B$ . For any  $1 \leq i \leq t-1$  and any fixed prefix  $v'_0, \dots, v'_{i-1}$ ,

$$\begin{aligned} & \mathbb{P}[v'_i = u, (v'_i, v'_{i-1}) \in E_0 \mid v'_0, \dots, v'_{i-1}] \\ & \leq \mathbf{1}[v'_{i-1} \in B] \cdot \frac{1}{d(v'_{i-1})} \\ & < \frac{1}{C}. \end{aligned} \tag{3}$$

Hence the probability that  $|\{1 \leq i \leq t-1 : v'_i = u, (v'_i, v'_{i-1}) \in E_0\}| \geq C$  is at most

$$\begin{aligned} & \binom{t-1}{C} \left(\frac{1}{C}\right)^C \\ & \leq \left(\frac{e(t-1)}{C}\right)^C \left(\frac{1}{C}\right)^C \\ & < \left(\frac{et}{C^2}\right)^C. \end{aligned}$$

We set  $C = \lceil 4\sqrt{t} q / \log q \rceil$ , where  $q = 2 + \log(1/\delta)/\sqrt{t} > 2$ . Notice that  $q/\log^2 q > 1/4$ . Then

$$C \log \left(\frac{C^2}{et}\right) \geq \frac{4\sqrt{t}q}{\log q} \log \left(\frac{16q^2}{e \log^2 q}\right) > \frac{4\sqrt{t}q}{\log q} \log(4q/e) > 4\sqrt{t}q > \log(1/\delta),$$

so

$$\left(\frac{et}{C^2}\right)^C < \delta.$$

Hence we have made  $\mathbb{P}[u \text{ fails} \mid v_0 = u] < \delta$  by choosing  $C = O(\sqrt{t}q / \log q)$ .  $\blacktriangleleft$

► **Theorem 16.** *We can simulate a  $t$ -step random walk on a simple undirected graph in the insertion-only model within error  $\varepsilon$  using  $O\left(n\sqrt{t} \cdot \frac{q}{\log q}\right)$  words of memory, where  $q = 2 + \frac{\log(1/\varepsilon)}{\sqrt{t}}$ .*

**Proof.** The theorem follows from Lemma 14 and Lemma 15 by setting  $\delta = \frac{\varepsilon}{2t}$ .  $\blacktriangleleft$

### 3.3 On graphs with multiple edges

When the undirected graph contains multiple edges, condition (3) in the proof of Lemma 15 may not hold, so we need to slightly modify our algorithm.



## 46:10 Simulating Random Walks on Graphs in the Streaming Model

We still maintain the multiset  $E_1$  of important arcs. Whether an arc is important will be determined by our algorithm. (This is different from the previous algorithm, where important arcs were simply defined as  $(u, v)$  with  $d(v) \leq C$ .) We will ensure that condition (3) still holds, i.e., for any  $u \in V$  and any fixed prefix of the random walk  $v_0, \dots, v_{i-1}$ ,

$$\mathbb{P}[(v_i, v_{i-1}) \notin E_1, \text{ and } v_i = u \mid v_0, \dots, v_{i-1}] < 1/C. \quad (4)$$

Note that there can be both important arcs and unimportant arcs from  $u$  to  $v$ . Let  $f(u, v)$  denote the number of undirected edges between  $u, v$ . Then there are  $f(u, v)$  arcs  $(u, v)$ . Suppose  $f_1(u, v)$  of these arcs are important, and  $f_0(u, v) = f(u, v) - f_1(u, v)$  of them are unimportant. Then we can rewrite condition (4) as

$$\frac{f_0(u, v_{i-1})}{d(v_{i-1})} < 1/C, \quad (5)$$

for every  $u, v_{i-1} \in V$ .

Similarly as before, we need to store the multiset  $E_1$  using only  $O(nC)$  words of space. And we need to sample with replacement  $C$  unimportant arcs  $a_{u,1}, \dots, a_{u,C}$  outgoing from  $u$ , for every  $u \in V$ . Finally we use the procedure SIMULATERANDOMWALK in Figure 3 to simulate a random walk.

The multiset  $E_1$  is determined as follows: For every vertex  $v \in V$ , we run Misra-Gries algorithm [16] on the sequence of all  $v$ 's neighbors. We will obtain a list  $L_v$  of at most  $C$  vertices, such that for every vertex  $u \notin L_v$ ,  $\frac{f(u,v)}{d(v)} < \frac{1}{C}$ . Moreover, we will get a frequency estimate  $A_v(u) > 0$  for every  $u \in L_v$ , such that  $0 \leq f(u, v) - A_v(u) < \frac{d(v)}{C}$ . Assuming  $A_v(u) = 0$  for  $u \notin L_v$ , we can satisfy condition (5) for all  $u \in V$  by setting  $f_1(u, v) = A_v(u)$ . Hence we have determined all the important arcs, and they can be stored in  $O(\sum_v |L_v|) = O(nC)$  words. To sample from the unimportant arcs, we simply insert the arcs discarded by Misra-Gries algorithm into the samplers. The pseudocode is given in Figure 4.

► **Lemma 17.** *After PROCESSINPUT (in Figure 4) finishes,  $|L_v| \leq C$ . For every  $u \in L_v$ ,  $0 \leq f(u, v) - A_v(u) \leq \frac{d(v)}{C+1}$ . For every  $u \notin L_v$ ,  $f(u, v) \leq \frac{d(v)}{C+1}$ .*

**Proof.** Every time the **for** loop in procedure INSERTARC finishes, the newly added vertex  $u$  must have been removed from  $L_v$ , so  $|L_v| \leq C$  still holds. Let  $W = \{w_1, \dots, w_{C+1}\}$  be the set of vertices in  $L_v$  before this **for** loop begins. Then for every  $u \in V$ ,  $f(u, v) - A_v(u)$  equals the number of times  $u$  is contained in  $W$  (assuming  $A_v(u) = 0$  for  $u \notin L_v$ ), which is at most  $\frac{1}{C+1} \sum_W |W| \leq \frac{d(v)}{C+1}$ . ◀

► **Corollary 18.** *Procedure PROCESSINPUT in Figure 4 computes the multiset  $E_1$  of important edges and stores it using  $O(nC)$  words. It also samples with replacement  $C$  unimportant arcs  $a_{u,1}, \dots, a_{u,C}$  outgoing from  $u$ , for every  $u \in V$ . Moreover,*

$$\frac{f_0(u, v)}{d(v)} < \frac{1}{C}$$

holds for every  $u, v \in V$ .

Now we analyze the failure probability of SIMULATERANDOMWALK (in Figure 3), similar to Lemma 15.

► **Lemma 19.** *We can choose integer parameter  $C = O\left(\sqrt{t} \cdot \frac{q}{\log q}\right)$ , where  $q = 2 + \frac{\log(1/\delta)}{\sqrt{t}}$ , so that  $\mathbb{P}[u \text{ fails} \mid v_0 = u] \leq \delta$  holds for every  $u \in V$ .*

```

procedure INSERTARC( $u, v$ )
   $d(v) \leftarrow d(v) + 1$ 
  if  $u \in L_v$  then
     $A_v(u) \leftarrow A_v(u) + 1$ 
  else
    Insert  $u$  into  $L_v$ 
     $A_v(u) \leftarrow 1$ 
    if  $|L_v| \geq C + 1$  then
      for  $w \in L_v$  do
        Feed arc  $(w, v)$  into  $w$ 's sampler
         $A_v(w) \leftarrow A_v(w) - 1$ 
        if  $A_v(w) = 0$  then
          Remove  $w$  from  $L_v$ 
        end if
      end for
    end if
  end if
end procedure
procedure PROCESSINPUT
  for  $u \in V$  do
     $d(u) \leftarrow 0$ 
    Initialize  $u$ 's sampler (initially empty) which maintains  $a_{u,1}, \dots, a_{u,C}$ 
    Initialize empty list  $L_u$ 
  end for
  for undirected edge  $(u, v)$  in the input stream do
    INSERTARC( $u, v$ )
    INSERTARC( $v, u$ )
  end for
   $E_1 \leftarrow \bigcup_{v \in V} \bigcup_{u \in L_v} \{A_v(u) \text{ copies of arc } (u, v)\}$  ▷ Multiset of important arcs
end procedure

```

■ **Figure 4** Pseudocode for processing the input stream (for undirected graphs with possibly multiple edges).

**Proof.** Let  $d_0(u) = \sum_{v \in V} f_0(u, v)$ . As before, we rewrite this probability as a sum over possible random walks. Here we distinguish between important and unimportant arcs. Denote  $s_i = \mathbf{1}[\text{step } (v_{i-1}, v_i) \text{ is along an important arc}]$ . Then for any  $u \in V$ ,

$$\begin{aligned}
& \mathbb{P}[u \text{ fails} \mid v_0 = u] \\
& \leq \mathbb{P}[u \text{ fails} \mid v_0 = u, \text{ arc } (v_0, v_1) \text{ is unimportant}] \\
& = \frac{d(u)}{d_0(u)} \sum_{k \leq t} \sum_{(v_0, \dots, v_k)} \sum_{s_1, \dots, s_k} \mathbf{1} \left[ v_0 = v_{k-1} = u, s_1 = s_k = 0, \right. \\
& \quad \left. |\{i : v_i = u, s_{i+1} = 0\}| = C + 1 \right] \prod_{i=0}^{k-1} \frac{f_{s_{i+1}}(v_i, v_{i+1})}{d(v_i)} \\
& = \sum_{k \leq t} \sum_{(v_0, \dots, v_{k-1})} \sum_{s_1, \dots, s_{k-1}} \mathbf{1} \left[ v_0 = v_{k-1} = u, s_1 = 0, \right. \\
& \quad \left. |\{i : v_i = u, s_{i+1} = 0\}| = C \right] \prod_{i=0}^{k-2} \frac{f_{s_{i+1}}(v_i, v_{i+1})}{d(v_i)}.
\end{aligned}$$

## 46:12 Simulating Random Walks on Graphs in the Streaming Model

Let  $v'_i = v_{k-1-i}$ ,  $s'_i = s_{k-i}$ . Then this sum equals

$$\begin{aligned} & \sum_{k \leq t} \sum_{(v'_0, \dots, v'_{k-1})} \sum_{(s'_1, \dots, s'_{k-1})} \mathbf{1} \left[ \begin{array}{l} v'_0 = v'_{k-1} = u, s'_{k-1} = 0, \\ |\{i : s'_i = 0, v'_i = u\}| = C \end{array} \right] \prod_{i=1}^{k-1} \frac{f_{s'_i}(v'_i, v'_{i-1})}{d(v'_{i-1})} \\ &= \mathbb{P}_{\text{random walk } (v'_0, \dots, v'_{t-1})} \left[ |\{i : v'_i = u, \text{arc } (v'_i, v'_{i-1}) \text{ is unimportant}\}| \geq C \mid v'_0 = u \right]. \end{aligned}$$

Notice that for any  $i$  and any fixed prefix  $v'_0, \dots, v'_{i-1}$ ,

$$\mathbb{P} \left[ v'_i = u, \text{arc } (v'_i, v'_{i-1}) \text{ is unimportant} \mid v'_0, v'_1, \dots, v'_{i-1} \right] = \frac{f_0(u, v'_{i-1})}{d(v'_{i-1})} < \frac{1}{C}$$

by Corollary 18. The rest of the proof is the same as in Lemma 15.  $\blacktriangleleft$

► **Theorem 20.** *We can simulate a random walk on an undirected graph with possibly multiple edges in the insertion-only model within error  $\varepsilon$  using  $O\left(n\sqrt{t} \cdot \frac{q}{\log q}\right)$  words of memory, where  $q = 2 + \frac{\log(1/\varepsilon)}{\sqrt{t}}$ .*

**Proof.** The theorem follows from Lemma 14 and Lemma 19 by setting  $\delta = \frac{\varepsilon}{2t}$ .  $\blacktriangleleft$

### 4 Turnstile model

In this section we consider the turnstile model where both insertion and deletion of edges can appear.

► **Lemma 21** ( $\ell_1$  sampler in the turnstile model, [9]). *Let  $f \in \mathbb{R}^n$  be a vector defined by a stream of updates to its coordinates of the form  $f_i \leftarrow f_i + \Delta$ , where  $\Delta$  can either be positive or negative. There is an algorithm which reads the stream and returns an index  $i \in [n]$  such that for every  $j \in [n]$ ,*

$$\mathbb{P}[i = j] = \frac{|f_j|}{\|f\|_1} + O(n^{-c}), \quad (6)$$

where  $c \geq 1$  is some arbitrary large constant. It is allowed to output FAIL with probability  $\delta$ , and in this case it will not output any index. The space complexity of this algorithm is  $O(\log^2 n \log(1/\delta))$  bits.

► **Remark.** For  $\varepsilon \ll 1/n$ , the  $O(n^{-c})$  error term in (6) can be reduced to  $O(\varepsilon^c)$  by running the  $\ell_1$  sampler on  $f \in \mathbb{R}^{\lceil 1/\varepsilon \rceil}$ , using  $O(\log^2(1/\varepsilon) \log(1/\delta))$  bits of space.

We will use the  $\ell_1$  sampler for sampling neighbors (with possibly multiple edges) in the turnstile model. The error term  $O(n^{-c})$  (or  $O(\varepsilon^c)$ ) in (6) can be ignored in the following discussion, by choosing sufficiently large constant  $c$  and scaling down  $\varepsilon$  by a constant.

#### 4.1 Directed graphs

► **Theorem 22.** *We can simulate a  $t$ -step random walk on a directed graph in the turnstile model within error  $\varepsilon$  using  $O(n(t + \log \frac{1}{\varepsilon}) \log^2 \max\{n, 1/\varepsilon\})$  bits of memory.*

**Proof.** For every  $u \in V$ , we run  $C' = 2t + 16 \log(2t/\varepsilon)$  independent  $\ell_1$  samplers each having failure probability  $\delta = 1/2$ . We use them to sample the outgoing edges of  $u$  (as in the algorithm of Theorem 8). By Chernoff bound, the probability that less than  $t$  samplers succeed is at most  $\varepsilon/(2t)$ .

We say a vertex  $u$  fails if  $u$  has less than  $t$  successful samplers, and  $u \in \{v_0, v_1, \dots, v_{t-1}\}$  (where  $v_0, v_1, \dots, v_t$  is the random walk). Then  $\mathbb{P}[u \text{ fails}] \leq \frac{\varepsilon}{2t} \mathbb{P}[u \in \{v_0, \dots, v_{t-1}\}]$ . By union bound,  $\mathbb{P}[\text{at least one vertex fails}] \leq \frac{\varepsilon}{2t} \sum_{u \in V} \mathbb{P}[u \in \{v_0, \dots, v_{t-1}\}] \leq \frac{\varepsilon}{2}$ . Hence, with probability  $1 - \frac{\varepsilon}{2}$ , every vertex  $u$  visited (except the last one) has at least  $t$  outgoing edges sampled, so our simulation can succeed. The space usage is  $O(nC' \log^2 \max\{n, 1/\varepsilon\} \log(1/\delta)) = O(n(t + \log \frac{1}{\varepsilon}) \log^2 \max\{n, 1/\varepsilon\})$  bits. ◀

## 4.2 Undirected graphs

We slightly modify the PROCESSINPUT procedure of our previous algorithm in Section 3.3. We will use the  $\ell_1$  heavy hitter algorithm in the turnstile model.

► **Lemma 23** ( $\ell_1$  heavy hitter, [6]). *Let  $f \in \mathbb{R}^n$  be a vector defined by a stream of updates to its coordinates of the form  $f_i \leftarrow f_i + \Delta$ , where  $\Delta$  can either be positive or negative. There is a randomized algorithm which reads the stream and returns a subset  $L \subseteq [n]$  such that  $i \in L$  for every  $|f_i| \geq \frac{\|f\|_1}{k}$ , and  $i \notin L$  for every  $|f_i| \leq \frac{\|f\|_1}{2k}$ . Moreover it returns a frequency estimate  $\tilde{f}_i$  for every  $i \in L$ , which satisfies  $0 \leq f_i - \tilde{f}_i \leq \frac{\|f\|_1}{2k}$ . The failure probability of this algorithm is  $O(n^{-c})$ . The space complexity is  $O(k \log^2 n)$  bits.*

► **Remark.** For  $\varepsilon \ll 1/n$ , the  $O(n^{-c})$  failure probability of this  $\ell_1$  heavy hitter algorithm can be reduced to  $O(\varepsilon^c)$  by running the algorithm on  $f \in \mathbb{R}^{\lceil 1/\varepsilon \rceil}$ , using  $O(k \log^2(1/\varepsilon))$  bits of space. In the following discussion, this failure probability can be ignored by making the constant  $c$  sufficiently large.

► **Theorem 24.** *We can simulate a  $t$ -step random walk on an undirected graph in the turnstile model within error  $\varepsilon$  using  $O(n(\sqrt{t} + \log \frac{1}{\varepsilon}) \log^2 \max\{n, 1/\varepsilon\})$  bits of memory.*

**Proof.** Similar to the previous insertion-only algorithm (in Figure 4), we perform two *arc updates*  $((u, v), \Delta)$ ,  $((v, u), \Delta)$  when we read an *edge update*  $((u, v), \Delta)$  from the stream.

For every  $u \in V$ , we run  $C' = 2C + 16 \log(2t/\varepsilon)$  independent  $\ell_1$  samplers each having failure probability  $\delta = 1/2$ , where  $C$  is the same constant as in the proof of Lemma 19 and Theorem 20. By Chernoff bound, the probability that less than  $C$  samplers succeed is at most  $\varepsilon/(2t)$ . For every arc update  $((u, v), \Delta)$ , we send update  $(v, \Delta)$  to  $u$ 's  $\ell_1$  sampler.

In addition, for every  $v \in V$ , we run  $\ell_1$  heavy hitter algorithm with  $k = C$ . For every arc update  $((u, v), \Delta)$ , we send update  $(u, \Delta)$  to  $v$ 's heavy hitter algorithm. In the end, we will get a frequency estimate  $A_v(u)$  for every  $u \in V$ , such that  $f(u, v) - \frac{d(v)}{C} \leq A_v(u) \leq f(u, v)$ . We then insert  $A_v(u)$  copies of arc  $(u, v)$  into  $E_1$  (the multiset of important arcs), and send update  $(v, -A_v(u))$  to  $u$ 's  $\ell_1$  sampler. Then we use the  $\ell_1$  samplers to sample unimportant arcs for every  $u$ .

As before, we use the procedure SIMULATERANDOMWALK (in Figure 3) to simulate the random walk. The analysis of the failure probability of the  $\ell_1$  samplers is the same as in Theorem 22. The analysis of the failure probability of procedure SIMULATERANDOMWALK is the same as in Lemma 19. The space usage of the algorithm is  $O(nC' \log^2 \max\{n, 1/\varepsilon\} \log \delta) = O(n(\sqrt{t} + \log \frac{1}{\varepsilon}) \log^2 \max\{n, 1/\varepsilon\})$  bits. ◀

## 5 Conclusion

We end our paper by discussing some related questions for future research.

- The output distribution of our insertion-only algorithm for undirected graphs is  $\varepsilon$ -close to the random walk distribution. What if the output is required to be perfectly random, i.e.,  $\varepsilon = 0$ ?
- For insertion-only simple undirected graphs, we proved an  $\Omega(n\sqrt{t})$ -bit space lower bound. Our algorithm uses  $O(n\sqrt{t} \log n)$  bits (for not too small  $\varepsilon$ ). Can we close the gap between the lower bound and the upper bound, as in the case of directed graphs?
- In the undirected version, suppose the starting vertex  $v_0$  is drawn from a distribution (for example, the stationary distribution of the graph) rather than being specified. Is it possible to obtain a better algorithm in this new setting? Notice that our proof of the  $\Omega(n\sqrt{t})$  lower bound does not work here, since it requires  $v_0$  to be specified.
- We required the algorithm to output all vertices on the random walk. If only the last vertex is required, can we get a better algorithm or prove non-trivial lower bounds?

---

## References

- 1 Kook Jin Ahn and Sudipto Guha. Linear programming in the semi-streaming model with application to the maximum matching problem. *Information and Computation*, 222:59–79, 2013. doi:10.1016/j.ic.2012.10.006.
- 2 Kook Jin Ahn, Sudipto Guha, and Andrew McGregor. Analyzing graph structure via linear measurements. In *Proceedings of the 23rd Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 459–467, 2012. doi:10.1137/1.9781611973099.40.
- 3 Reid Andersen, Fan Chung, and Kevin Lang. Using pagerank to locally partition a graph. *Internet Mathematics*, 4(1):35–64, 2007. doi:10.1080/15427951.2007.10129139.
- 4 Reid Andersen and Yuval Peres. Finding sparse cuts locally using evolving sets. In *Proceedings of the 41st Annual ACM Symposium on Theory of Computing (STOC)*, pages 235–244, 2009. doi:10.1145/1536414.1536449.
- 5 Moses Charikar, Liadan O’Callaghan, and Rina Panigrahy. Better streaming algorithms for clustering problems. In *Proceedings of the 35th Annual ACM Symposium on Theory of Computing (STOC)*, pages 30–39, 2003. doi:10.1145/780542.780548.
- 6 Graham Cormode and Shan Muthukrishnan. An improved data stream summary: the count-min sketch and its applications. *Journal of Algorithms*, 55(1):58–75, 2005. doi:10.1016/j.jalgor.2003.12.001.
- 7 Atish Das Sarma, Sreenivas Gollapudi, and Rina Panigrahy. Estimating pagerank on graph streams. *Journal of the ACM (JACM)*, 58(3):13, 2011. doi:10.1145/1970392.1970397.
- 8 Leah Epstein, Asaf Levin, Julián Mestre, and Danny Segev. Improved Approximation Guarantees for Weighted Matching in the Semi-streaming Model. *SIAM Journal on Discrete Mathematics*, 25(3):1251–1265, 2011. doi:10.1137/100801901.
- 9 Rajesh Jayaram and David P. Woodruff. Perfect Lp Sampling in a Data Stream. In *Proceedings of the 59th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 544–555, 2018. doi:10.1109/FOCS.2018.00058.
- 10 Mark Jerrum and Alistair Sinclair. Approximating the permanent. *SIAM Journal on Computing*, 18(6):1149–1178, 1989. doi:10.1137/0218077.
- 11 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. doi:10.1016/0304-3975(86)90174-X.

- 12 Michael Kapralov. Better bounds for matchings in the streaming model. In *Proceedings of the 24th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1679–1697, 2013. doi:10.1137/1.9781611973105.121.
- 13 Michael Kapralov, Yin Tat Lee, Cameron Musco, Christopher Musco, and Aaron Sidford. Single pass spectral sparsification in dynamic streams. *SIAM Journal on Computing*, 46(1):456–477, 2017. doi:10.1137/141002281.
- 14 Jonathan A. Kelner and Alex Levin. Spectral sparsification in the semi-streaming setting. *Theory of Computing Systems*, 53(2):243–262, 2013. doi:10.1007/s00224-012-9396-1.
- 15 Peter Bro Miltersen, Noam Nisan, Shmuel Safra, and Avi Wigderson. On Data Structures and Asymmetric Communication Complexity. *Journal of Computer and System Sciences*, 57(1):37–49, 1998. doi:10.1006/jcss.1998.1577.
- 16 J. Misra and David Gries. Finding repeated elements. *Science of Computer Programming*, 2(2):143–152, 1982. doi:10.1016/0167-6423(82)90012-0.
- 17 Omer Reingold. Undirected connectivity in log-space. *Journal of the ACM (JACM)*, 55(4):17, 2008. doi:10.1145/1391289.1391291.
- 18 Daniel A. Spielman and Shang-Hua Teng. A Local Clustering Algorithm for Massive Graphs and Its Application to Nearly Linear Time Graph Partitioning. *SIAM Journal on Computing*, 42(1):1–26, 2013. doi:10.1137/080744888.





# On the Complexity of Symmetric Polynomials

Markus Bläser

Department of Computer Science, Saarland University, Saarland Informatics Campus,  
Saarbrücken, Germany  
mblaeser@cs.uni-saarland.de

Gorav Jindal<sup>1</sup>

Department of Computer Science, Aalto University, Espoo, Finland  
gorav.jindal@gmail.com

---

## Abstract

---

The fundamental theorem of symmetric polynomials states that for a symmetric polynomial  $f_{\text{Sym}} \in \mathbb{C}[x_1, x_2, \dots, x_n]$ , there exists a unique “witness”  $f \in \mathbb{C}[y_1, y_2, \dots, y_n]$  such that  $f_{\text{Sym}} = f(e_1, e_2, \dots, e_n)$ , where the  $e_i$ ’s are the elementary symmetric polynomials.

In this paper, we study the arithmetic complexity  $L(f)$  of the witness  $f$  as a function of the arithmetic complexity  $L(f_{\text{Sym}})$  of  $f_{\text{Sym}}$ . We show that the arithmetic complexity  $L(f)$  of  $f$  is bounded by  $\text{poly}(L(f_{\text{Sym}}), \deg(f), n)$ . To the best of our knowledge, prior to this work only exponential upper bounds were known for  $L(f)$ . The main ingredient in our result is an algebraic analogue of Newton’s iteration on power series. As a corollary of this result, we show that if  $\text{VP} \neq \text{VNP}$  then there exist symmetric polynomial families which have super-polynomial arithmetic complexity.

Furthermore, we study the complexity of testing whether a function is symmetric. For polynomials, this question is equivalent to arithmetic circuit identity testing. In contrast to this, we show that it is hard for Boolean functions.

**2012 ACM Subject Classification** Theory of computation  $\rightarrow$  Algebraic complexity theory

**Keywords and phrases** Symmetric Polynomials, Arithmetic Circuits, Arithmetic Complexity, Power Series, Elementary Symmetric Polynomials, Newton’s Iteration

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2019.47

## 1 Introduction

Lipton and Regan [10] ask the question whether understanding the arithmetic complexity of symmetric polynomials is enough to understand the arithmetic complexity of all polynomials. We here answer this question in the affirmative. The fundamental theorem of symmetric polynomials establishes a bijection between symmetric polynomials and arbitrary polynomials. It states that for every symmetric polynomial  $f_{\text{Sym}} \in \mathbb{C}[x_1, x_2, \dots, x_n]$ , there exists a unique polynomial  $f \in \mathbb{C}[y_1, y_2, \dots, y_n]$  such that  $f_{\text{Sym}} = f(e_1, e_2, \dots, e_n)$ , where the  $e_i$ ’s are the elementary symmetric polynomials. We prove that the arithmetic circuit complexity of  $f$  and  $f_{\text{Sym}}$  are polynomially related.

An arithmetic circuit  $C$  is a directed acyclic graph with the following kind of nodes (gates):

---

<sup>1</sup> Supported by European Research Council (ERC) under the European Union’s Horizon 2020 research and innovation program (grant agreement No 759557) and by Academy of Finland, under grant number 310415. This work was done while the author was a graduate student at Saarland University and the Max-Planck-Institut für Informatik.



## 47:2 Complexity of Symmetric Polynomials

- Nodes with in-degree zero labeled by variables or scalars, these are called input gates.
- Nodes labeled by addition (+), subtraction (−) or multiplication (×) gates, these gates have in-degree two and unbounded out-degree.
- Nodes with out-degree zero (it can be an input, +, − or × gate), these are called output gates.

Each gate of such a circuit computes a multivariate polynomial in the following way:

- Input gates compute the polynomial by which they are labeled.
- A  $\circ$  gate  $g$  computes the polynomial  $g_1 \circ g_2$ , if the children gates of  $g$  compute the polynomials  $g_1$  and  $g_2$ , here  $\circ \in \{+, -, \times\}$ .

If the output gates of  $C$  compute the polynomials  $g_1, g_2, \dots, g_t$  then we say that  $C$  computes the set  $\{g_1, g_2, \dots, g_t\}$  of polynomials. In the literature, it is usually assumed that any arithmetic circuit  $C$  has a unique output gate and thus  $C$  computes a single multivariate polynomial. The *size* of an arithmetic circuit  $C$  is defined as the number of gates in  $C$ .

We can naturally model computations over a field by arithmetic circuits and thus the study of arithmetic circuits is essential in studying the computational complexity in algebraic models of computation. Valiant [15] defined the complexity classes  $\mathbf{VP}$  and  $\mathbf{VNP}$  as algebraic analogues of the classes  $\mathbf{P}$  and  $\mathbf{NP}$ . In algebraic complexity theory, complexity classes such as  $\mathbf{VP}$  and  $\mathbf{VNP}$  are defined as sets of polynomial families. For the precise definitions of  $\mathbf{VP}$  and  $\mathbf{VNP}$ , see section A in the appendix. For a polynomial  $f$ ,  $L(f)$  is defined as the size of the smallest circuit computing  $f$ . We use the same notation  $L(S)$  to denote the size of the smallest circuit computing a set of polynomials  $S$ . There is also a notion of oracle complexity  $L^g(f)$  of a polynomial  $f$  with respect to some polynomial  $g$ , see Definition 9.

Let  $\mathfrak{S}_n$  be the symmetric group defined as the set of all permutations of the set  $\{1, 2, \dots, n\}$ . A Boolean function  $f : \{0, 1\}^n \rightarrow \{0, 1\}$  is said to be *symmetric* if  $f(x_1, x_2, \dots, x_n) = f(x_{\sigma(1)}, x_{\sigma(2)}, \dots, x_{\sigma(n)})$  for all  $(x_1, x_2, \dots, x_n) \in \{0, 1\}^n, \sigma \in \mathfrak{S}_n$ . It is easy to see that all symmetric Boolean functions can be computed by constant depth threshold circuits, that is, are contained in the class  $\mathbf{TC}^0$ . The notion of symmetric polynomials can also be defined similarly. It is natural to ask whether symmetric polynomials can also be computed efficiently, *i.e.*, whether the arithmetic complexity of symmetric polynomials is also small? This is the question we study in this paper.

### 1.1 Previous Work

It is well known that for every symmetric polynomial  $g \in \mathbb{C}[x_1, x_2, \dots, x_n]$ , there exists a unique polynomial  $f \in \mathbb{C}[x_1, x_2, \dots, x_n]$  such that  $g = f(e_1, e_2, \dots, e_n)$ . Here,  $e_1, e_2, \dots, e_n$  denote the elementary symmetric polynomials in  $x_1, x_2, \dots, x_n$ . For an arbitrary polynomial  $f \in \mathbb{C}[x_1, x_2, \dots, x_n]$ , let  $f_{\text{Sym}}$  defined by  $f_{\text{Sym}} \stackrel{\text{def}}{=} f(e_1, e_2, \dots, e_n)$  be the symmetric polynomial corresponding to  $f$  (see Section 2). We want to study the relation between the complexities  $L(f)$  and  $L(f_{\text{Sym}})$ . The question was studied and partially solved in [7, 6, 10]. More specifically, the following theorems were proved in [7, 6, 10].

► **Theorem 1** (Theorem 1 in [7]). *For any polynomial  $f \in \mathbb{C}[x_1, x_2, \dots, x_n]$ ,  $L(f) \leq \Delta(n)L(f_{\text{Sym}}) + 2$ , where  $\Delta(n) \leq 4^n(n!)^2$ .*

Whereas [7] showed the bound on  $L(f)$  for exact computation, [10] investigated a related problem of approximating the value of  $f$  at a given point by using an arithmetic circuit computing  $f_{\text{Sym}}$ .

► **Theorem 2** ([10]). *For any polynomial  $f \in \mathbb{Q}[x_1, x_2, \dots, x_n]$ , there is an algorithm that computes the value  $f(a)$  within  $\epsilon$  in time  $L(f_{\text{Sym}}) + \text{poly}(\log \|a\|, n, \log \frac{1}{\epsilon})$  for any  $a \in \mathbb{Q}^n$ .*

Note that Theorem 2 does not compute a circuit for  $f$  but only gives an algorithm to approximate the value of  $f$  at a given point. The results in [6] were in a much more general setting. [6] studied the in-variance under general finite matrix groups, not just under  $\mathfrak{S}_n$  as we do in this paper. By specializing the theorems in [6] for the finite matrix group  $\mathfrak{S}_n$ , we get the following result.

► **Theorem 3** ([6]). *For any polynomial  $f \in \mathbb{C}[x_1, x_2, \dots, x_n]$ , we have  $L(f) \leq ((n+1)!)^6 L(f_{\text{Sym}})$ .*

The upper bound in [6] (Theorem 3) is worse than that of [7] (Theorem 1) but this is to be expected because [6] solves a more general problem.

## 1.2 Our results

It is easy to see that  $L(f_{\text{Sym}}) \leq L(f) + n^{O(1)}$  (see [10]). All the exact bounds (with respect to  $L(f_{\text{Sym}})$ ) on  $L(f)$  above are exponential. It was left as an open question in [10] whether  $L(f)$  can be bounded polynomially with respect to  $L(f_{\text{Sym}})$ . In this paper, we demonstrate that  $L(f)$  can be polynomially bounded in terms of  $L(f_{\text{Sym}})$ . In whatever follows, the complexity notation  $\tilde{O}$  hides poly-logarithmic factors. The following Theorem 4 is the main contribution of this paper.

► **Theorem 4.** *For any polynomial  $f \in \mathbb{C}[x_1, x_2, \dots, x_n]$  of degree  $d$  with  $d_{\text{Sym}} \stackrel{\text{def}}{=} \deg(f_{\text{Sym}})$ , we have the following upper bounds on  $L^{f_{\text{Sym}}}(f)$  and  $L(f)$ :*

$$\begin{aligned} L^{f_{\text{Sym}}}(f) &\leq \tilde{O}(n^3 \cdot d^2 \cdot d_{\text{Sym}}), \\ L(f) &\leq \tilde{O}(d^2 L(f_{\text{Sym}}) + d^2 n^2). \end{aligned}$$

► **Remark.** It can be shown that  $d_{\text{Sym}} \leq dn$  and  $d \leq d_{\text{Sym}}$ . Thus Theorem 4 implies that  $L^{f_{\text{Sym}}}(f) \leq \tilde{O}(n^4 \cdot d^3)$ .

From Theorem 4, it easily follows that there exist families of symmetric polynomials of super-polynomial arithmetic complexity (assuming  $\text{VP} \neq \text{VNP}$ ).

In addition, we also consider the following problems:

1. SFT (symmetric function testing)
2. SPT (symmetric polynomial testing)

► **Problem 5** (SFT). *Given a Boolean circuit  $C$  computing the Boolean function  $f(x_1, x_2, \dots, x_n)$ , check if  $f$  is symmetric, that is, if  $f(x_1, x_2, \dots, x_n) = f(x_{\sigma(1)}, x_{\sigma(2)}, \dots, x_{\sigma(n)})$  for all  $\sigma \in \mathfrak{S}_n$ ?*

► **Problem 6** (SPT). *Given an arithmetic circuit  $C$  computing the polynomial  $f(x_1, x_2, \dots, x_n)$ , check if  $f$  is a symmetric polynomial?*

Let CSAT be the problem of deciding whether a given Boolean circuit has a satisfying assignment, that is, computes a non-zero function (see [8]). ACIT is the problem of deciding whether the polynomial computed by a given arithmetic circuit is zero (see [1]). We prove the following results on the complexity of SPT and SFT.

► **Lemma 7.** *SFT and CSAT are polynomial time Turing reducible to each other, i.e.,  $\text{SFT} \leq_{\text{P}}^{\text{T}} \text{CSAT}$  and  $\text{CSAT} \leq_{\text{P}}^{\text{T}} \text{SFT}$ .*

► **Lemma 8.** *SPT and ACIT are polynomial time many one reducible to each other, i.e.,  $\text{SPT} \leq_{\text{P}} \text{ACIT}$  and  $\text{ACIT} \leq_{\text{P}} \text{SPT}$ .*

## 47:4 Complexity of Symmetric Polynomials

In light of above results, we notice the following contrasting situations in Boolean and algebraic models of computation:

- All symmetric Boolean functions are easy to compute but the problem of deciding the symmetry of a Boolean function is hard.
- There exist families of symmetric polynomials which are hard to compute (assuming  $VP \neq VNP$ ) but deciding the symmetry of a polynomial is easy.

### 1.3 Proof ideas

The main proof idea is much easier to demonstrate in the case of  $n = 2$ . Let  $B(y)$  be the following uni-variate polynomial in  $y$  with coefficients in  $\mathbb{C}[x_1, x_2]$ :

$$B(y) \stackrel{\text{def}}{=} y^2 - (x_1 + x_2)y + x_1x_2.$$

Note that the roots of  $B(y)$  are  $x_1, x_2$ . Hence we have the following equalities:

$$x_1 = \frac{x_1 + x_2 + \sqrt{(x_1 + x_2)^2 - 4x_1x_2}}{2},$$

$$x_2 = \frac{x_1 + x_2 - \sqrt{(x_1 + x_2)^2 - 4x_1x_2}}{2}.$$

We use the symbols  $e_1, e_2$  for the elementary symmetric polynomials:  $e_1 \stackrel{\text{def}}{=} (x_1 + x_2)$  and  $e_2 \stackrel{\text{def}}{=} x_1x_2$ . Thus we have the following equalities:

$$x_1 = \frac{e_1 + \sqrt{e_1^2 - 4e_2}}{2}, \tag{1}$$

$$x_2 = \frac{e_1 - \sqrt{e_1^2 - 4e_2}}{2}. \tag{2}$$

Let  $f_{\text{Sym}} \in \mathbb{C}[x_1, x_2]$  be a symmetric polynomial and  $\deg(f) = d$ .

If we substitute the above radical expressions (in Equation 1 and Equation 2) for  $x_1$  and  $x_2$  in  $f_{\text{Sym}}(x_1, x_2)$ , then we obtain  $f(e_1, e_2)$ . But unfortunately, we can not perform these kind of substitutions in our model of computation. This is because we cannot compute expressions of the form  $\sqrt{e_1^2 - 4e_2}$  with arithmetic circuits.

If we use the substitution  $e_2 \leftarrow e_2 - 1$  in Equation 1 and Equation 2 and thereafter substitute  $x_1$  and  $x_2$  in  $f_{\text{Sym}}(x_1, x_2)$ , we shall obtain  $f(e_1, e_2 - 1)$ . The degree of  $f(e_1, e_2 - 1)$  is also bounded by  $d$ . Even by using this substitution, the expressions in Equation 1 and Equation 2 cannot be computed by arithmetic circuits. But this substitution allows us to use Taylor expansion on  $\sqrt{e_1^2 - 4(e_2 - 1)}$  to obtain a power series in  $e_1, e_2$ . Since  $f(e_1, e_2 - 1)$  has degree at most  $d$ , we only need to substitute truncations of degree  $d$  of these Taylor series to obtain  $f(e_1, e_2 - 1)$  (also some additional junk terms which can be removed efficiently) and subsequently use the substitution  $e_2 \leftarrow e_2 + 1$  to obtain  $f(e_1, e_2)$ .

This method works for two variables. It can be extended to work for at most four variables because it is well known that polynomials of degree more than four are not solvable by radicals (see e.g. Section 15.9 in [2]). To make this idea work in general, we shall substitute  $e_n$  by  $e_n + (-1)^{n-1}$  and then compute degree  $d$  truncation of roots of  $B(y)$  using Newton's iteration.

## 1.4 Organization

Section 2 introduces elementary symmetric polynomials and also formally states the fundamental theorem of symmetric polynomials. We also state a folklore result about computing the homogeneous components of a polynomial. Section 3 describes the complexity of the problems SFT and SPT. Section 4 describes the main contribution of this paper, we use the classical Newton's iteration to prove Theorem 4 in this section. As an easy consequence of Theorem 4, Section 5 proves that there exist hard symmetric polynomial families (assuming  $VP \neq VNP$ ).

## 2 Preliminaries

### 2.1 Notation and background

For a positive integer  $n$ , we use the notation  $[n]$  to denote the set  $\{1, 2, \dots, n\}$ . Similarly,  $[[n]]$  is used to denote the set  $\{0, 1, 2, \dots, n\}$ . Now we formally define the notion of oracle computations [5].

► **Definition 9** ([5]). The oracle complexity  $L^g(f)$  of a polynomial  $f \in \mathbb{F}[x_1, x_2, \dots, x_n]$  with respect to the oracle polynomial  $g$  is the minimum number of arithmetic operations  $+$ ,  $-$ ,  $\times$  and evaluations of  $g$  (at previously computed values) that are sufficient to compute  $f$  from the indeterminates  $x_i$  and constants in  $\mathbb{F}$ .

► **Definition 10.** The  $i^{\text{th}}$  elementary symmetric polynomial  $e_i^n$  in  $n$  variables  $x_1, x_2, \dots, x_n$  is defined as the following polynomial:

$$e_i^n \stackrel{\text{def}}{=} \sum_{1 \leq j_1 < j_2 < \dots < j_i \leq n} x_{j_1} \cdot x_{j_2} \cdot \dots \cdot x_{j_i}.$$

For an arbitrary polynomial  $f \in \mathbb{F}[x_1, x_2, \dots, x_n]$ , we define the polynomial  $f_{\text{Sym}}$  as:

$$f_{\text{Sym}} \stackrel{\text{def}}{=} f(e_1^n, e_2^n, \dots, e_n^n). \quad (3)$$

Whenever  $n$  is clear from the context, we use the notation  $e_i$  to denote the  $i^{\text{th}}$  elementary symmetric polynomial  $e_i^n$ . Note that  $f_{\text{Sym}}$  is a symmetric polynomial. So Equation 3 is a method to create symmetric polynomials. The fundamental theorem of symmetric polynomials states that Equation 3 is the only way to create symmetric polynomials.

► **Theorem 11** (see [3]). *If  $g \in \mathbb{C}[x_1, x_2, \dots, x_n]$  is a symmetric polynomial, then there exists a unique polynomial  $f \in \mathbb{C}[y_1, y_2, \dots, y_n]$  such that  $g = f(e_1^n, e_2^n, \dots, e_n^n)$ .*

Theorem 11 states that every symmetric polynomial  $g$  can be uniquely written as  $f_{\text{Sym}}$  for some  $f$ . Thus in whatever follows, we always use the notation of the kind  $f_{\text{Sym}}$  to denote a symmetric polynomial.

### 2.2 Basic tools

Suppose we have a circuit  $C$  computing a polynomial  $f \in \mathbb{C}[x_1, x_2, \dots, x_n]$  of degree  $d$ . It might be the case that  $f$  is not homogeneous. For some applications, it might be better to work with homogeneous polynomials. So we want to know if there exist “small” circuits also for the homogeneous components of  $f$ . For a polynomial  $f$ ,  $f^{[m]}$  is used to denote the degree  $m$  homogeneous component of  $f$ . The following Lemma 12 proves that the homogeneous components of  $f$  also have “small” arithmetic circuits.

## 47:6 Complexity of Symmetric Polynomials

► **Lemma 12** (Folklore). *Let  $f \in \mathbb{C}[x_1, x_2, \dots, x_n]$  be a polynomial with  $d = \deg(f)$ . For any  $0 \leq m \leq d$ , we have:  $L^f(f^{[m]}) \leq O(nd)$ .*

**Proof.** For a fresh indeterminate  $y$ , consider the polynomial  $f(yx_1, yx_2, \dots, yx_n)$ . We consider  $f(yx_1, yx_2, \dots, yx_n)$  as a uni-variate polynomial in  $y$  of degree  $d$ , with coefficients in  $\mathbb{C}[x_1, x_2, \dots, x_n]$ . We observe that for any  $0 \leq m \leq d$ , the coefficient of  $y^m$  in  $f(yx_1, yx_2, \dots, yx_n)$  is  $f^{[m]}$ . Let  $\alpha_1, \alpha_2, \dots, \alpha_{d+1}$  be  $d+1$  distinct constants in  $\mathbb{C}$ . By interpolation, we know that for any  $0 \leq m \leq d$ , the coefficient  $f^{[m]}$  of  $y^m$  is a  $\mathbb{C}$ -linear combination of  $d+1$  evaluations  $f(\alpha_i x_1, \alpha_i x_2, \dots, \alpha_i x_n)$  of  $f(yx_1, yx_2, \dots, yx_n)$  at  $\alpha_i \in \{\alpha_1, \alpha_2, \dots, \alpha_{d+1}\}$ . Formally, for any  $0 \leq m \leq d$  we have:

$$f^{[m]} \in \langle \{f(\alpha_i x_1, \alpha_i x_2, \dots, \alpha_i x_n) \mid \alpha_i \in \{\alpha_1, \alpha_2, \dots, \alpha_{d+1}\}\} \rangle. \quad (4)$$

It is easy to observe that  $L^f(f(\alpha_i x_1, \alpha_i x_2, \dots, \alpha_i x_n)) = O(n)$ . Equation 4 implies that  $L^f(f^{[m]}) \leq O(nd)$ . ◀

Lemma 12 implies that  $L(f^{[m]}) \leq O(L(f) \cdot n \cdot d)$ , this bound depends on the degree  $d$  of  $f$ . But if we do not care about the oracle complexity, the following bound (independent of degree of  $f$ ) on  $L(f^{[m]})$  is known.

► **Lemma 13** ([13, 14]). *Let  $f \in \mathbb{C}[x_1, x_2, \dots, x_n]$  be a polynomial and  $m$  be a non-negative integer. Then we have:*

$$L(\{f^{[0]}, f^{[1]}, \dots, f^{[m]}\}) \leq O(m^2 L(f)).$$

### 3 Complexity of SFT and SPT

Here we prove Lemma 7 and Lemma 8.

**Proof of Lemma 7.** Given a Boolean circuit  $C$ , we want to check if the function  $f(x_1, x_2, \dots, x_n)$  computed by  $C$  is symmetric. As the permutation group  $\mathfrak{S}_n$  is generated by two permutations  $\sigma \stackrel{\text{def}}{=} (1, 2)$  and  $\pi \stackrel{\text{def}}{=} (1, 2, \dots, n)$  [4], it is necessary and sufficient to check if the given function  $f$  is invariant under these two permutations of variables. Thus we define the following Boolean functions:

$$\begin{aligned} g(x_1, x_2, \dots, x_n) &\stackrel{\text{def}}{=} f(x_{\sigma(1)}, x_{\sigma(2)}, \dots, x_{\sigma(n)}), \\ h(x_1, x_2, \dots, x_n) &\stackrel{\text{def}}{=} f(x_{\pi(1)}, x_{\pi(2)}, \dots, x_{\pi(n)}). \end{aligned}$$

Now note that the equality of two Boolean variables  $x, y$  can be checked by the following equality gadget:

$$(x \stackrel{?}{=} y) = (\neg x \vee y) \wedge (x \vee \neg y).$$

Thus we only need to check if both  $(\neg f \vee g) \wedge (f \vee \neg g)$  and  $(\neg f \vee h) \wedge (f \vee \neg h)$  are tautologies (always equal to 1). This can be checked by two oracle calls to CSAT. Thus  $\text{SFT} \leq_{\mathbb{P}}^T \text{CSAT}$ .

Now we prove the other direction. Given a Boolean circuit  $C$ , we want to check if the function  $f(x_1, x_2, \dots, x_n)$  computed by  $C$  is always zero. First we make an oracle call to SFT to check if  $f$  is symmetric. If  $f$  is not symmetric then obviously  $f$  is a non-zero function because the zero function is trivially symmetric. Thus we can assume  $f$  to be symmetric. Now we ask the SFT oracle if the function  $h \stackrel{\text{def}}{=} f \wedge x_1$  is symmetric? If  $f$  was the zero

function then so is  $h$ , therefore SFT oracle will answer that  $h$  is symmetric. So if SFT oracle answers  $h$  to be non-symmetric then obviously  $f$  was non-zero. If  $h$  also turns out to be symmetric then we know that:

$$\forall (a_1, a_2, \dots, a_n) \in \{0, 1\}^n : f \wedge a_1 = f \wedge a_2 = \dots = f \wedge a_n. \quad (5)$$

Suppose  $f$  evaluated to 1 on a point  $(a_1, a_2, \dots, a_n) \notin \{(0, 0, \dots, 0), (1, 1, \dots, 1)\}$ . This means that there exists  $(a_1, a_2, \dots, a_n) \in \{0, 1\}^n$  such that  $f(a_1, a_2, \dots, a_n) = 1$  with  $a_i = 1, a_j = 0$  for some  $i, j \in [n]$ . Then obviously we have  $f(a_1, a_2, \dots, a_n) \wedge a_i = 1$  and  $f(a_1, a_2, \dots, a_n) \wedge a_j = 0$ . Hence Equation 5 can not to be true. Thus  $f$  can only be non-zero on the set  $\{(0, 0, \dots, 0), (1, 1, \dots, 1)\}$ . The value of  $f$  at both these points can be checked manually to check whether  $f$  is the zero function or not. Therefore  $\text{CSAT} \leq_{\text{P}}^{\text{T}} \text{SFT}$ . ◀

**Proof of Lemma 8.** Given an arithmetic circuit  $C$ , we want to check if the polynomial  $f(x_1, x_2, \dots, x_n)$  computed by  $C$  is symmetric.

As in the proof of the Lemma 7, we use the fact that permutation group  $\mathfrak{S}_n$  is generated by two permutations  $\sigma \stackrel{\text{def}}{=} (1, 2)$  and  $\pi \stackrel{\text{def}}{=} (1, 2, \dots, n)$ . It is necessary and sufficient to check if the given polynomial  $f$  is invariant under these two permutations of variables. Analogous to the proof of the Lemma 7, we define the following polynomials:

$$g(x_1, x_2, \dots, x_n) \stackrel{\text{def}}{=} f(x_{\sigma(1)}, x_{\sigma(2)}, \dots, x_{\sigma(n)}),$$

$$h(x_1, x_2, \dots, x_n) \stackrel{\text{def}}{=} f(x_{\pi(1)}, x_{\pi(2)}, \dots, x_{\pi(n)}).$$

Thus  $f$  is symmetric iff  $f - g = f - h = 0$ . Consider the polynomial  $F = y(f - g) + z(f - h)$ , here  $y, z$  are fresh variables. Thus  $f$  is symmetric iff  $F$  is the zero polynomial. Hence  $\text{SPT} \leq_{\text{P}} \text{ACIT}$ .

Now we prove the reverse direction. Given an arithmetic circuit  $C$ , we want to check if the polynomial  $f(x_1, x_2, \dots, x_n)$  computed by  $C$  is the zero polynomial or not. Consider the polynomial  $G \stackrel{\text{def}}{=} f(x_1^2, x_2^2, \dots, x_n^2) \cdot x_1$ . We know that  $f$  is non-zero iff  $G$  is non-zero. Suppose that  $G \neq 0$ . Now observe that in every monomial  $\mathcal{M}$  of  $G$ , the degree of  $x_1$  in  $\mathcal{M}$  is odd and the degrees of the other variables  $x_2, \dots, x_n$  in  $\mathcal{M}$  are even. Now consider the polynomial  $H \stackrel{\text{def}}{=} G(x_{\sigma(1)}, x_{\sigma(2)}, \dots, x_{\sigma(n)})$  where  $\sigma \stackrel{\text{def}}{=} (1, 2)$ . In every monomial  $\mathcal{M}'$  of  $H$ , the degree of  $x_2$  in  $\mathcal{M}'$  is odd and the degrees of the other variables  $x_1, x_3, \dots, x_n$  in  $\mathcal{M}'$  are even. Thus  $H \neq G$ . Hence if  $G$  is non-zero then  $G$  is not symmetric because it is not invariant under the permutation  $\sigma \stackrel{\text{def}}{=} (1, 2)$ . Thus  $G$  is symmetric iff  $f = 0$ . Hence  $\text{ACIT} \leq_{\text{P}} \text{SPT}$ . ◀

## 4 Main algorithm

### 4.1 Roots as power series

Let  $F(y) = F(y, u_1, u_2, \dots, u_n) = y^n + f_1(u_1, u_2, \dots, u_n)y^{n-1} + \dots + f_n(u_1, u_2, \dots, u_n)$  be a monic square-free polynomial in variables  $y$  and  $u_1, u_2, \dots, u_n$ , here  $f_i \in \mathbb{C}[u_1, u_2, \dots, u_n]$ . Let  $A(u_1, u_2, \dots, u_n)$  be a root of  $F$  with respect to  $y$ . The root is usually an algebraic function in  $u_1, u_2, \dots, u_n$  but not a power series. The following Lemma 14 formalizes a sufficient condition when roots of  $F(y)$  can be expressed as power series in  $u_1, u_2, \dots, u_n$ .

► **Lemma 14** (Condition A in [12]). *Let  $F(y, u_1, u_2, \dots, u_n)$  be square free and monic with respect to  $y$ . If  $F(y, 0, 0, \dots, 0)$  has no multiple root (as a uni-variate polynomial in  $y$ ) then the roots  $A_i(u_1, u_2, \dots, u_n)$  of  $F(y, u_1, u_2, \dots, u_n)$  can be expanded into power series in  $u_1, u_2, \dots, u_n$ .*



---

**Algorithm 1** Newton's Method.

---

**Input:** A square free monic polynomial  $F(y) = F(y, u_1, u_2, \dots, u_n) \in \mathbb{C}[u_1, u_2, \dots, u_n][y]$  with respect to  $y$  of degree  $n$  such that  $F(y, 0, 0, \dots, 0)$  has  $n$  simple roots. A positive integer  $d$  with  $d = 2^\ell$  for some  $\ell \in \mathbb{N}$ . We assume that  $A_1, A_2, \dots, A_n \in \mathbb{C}[[u_1, u_2, \dots, u_n]]$  are the roots of  $F(y)$ .

**Output:** Degree  $d$  truncations  $A_1^{(\ell)}, A_2^{(\ell)}, \dots, A_n^{(\ell)}$  of the  $n$  roots  $(A_1, A_2, \dots, A_n)$  of  $F(y)$ , that is,  $A_i^{(\ell)} \equiv A_i \pmod{I^d}$  with  $I \stackrel{\text{def}}{=} (u_1, u_2, \dots, u_n)$  for all  $i \in [n]$ .

1:  $\{\alpha_1, \alpha_2, \dots, \alpha_n\} \leftarrow$  Roots of  $F(y, 0, 0, \dots, 0)$ .

2: **for**  $1 \leq i \leq n$  **do**

3:      $A_i^{(0)} \leftarrow \alpha_i$ .

4:     **for**  $0 \leq k \leq \ell - 1$  **do**

5:          $A_i^{(k+1)} \leftarrow A_i^{(k)} - \frac{F(A_i^{(k)})}{F'(A_i^{(k)})}$ .

6:     **end for**

7: **end for**

8: **return**  $A_1^{(\ell)}, A_2^{(\ell)}, \dots, A_n^{(\ell)}$ .

---

As stated above, we are interested in the following special case:

$$F(y, e_1, e_2, \dots, e_n) = y^n - e_1 y^{n-1} + \dots + (-1)^n e_n.$$

This  $F$  is being considered as a uni-variate polynomial in  $y$  over the power series ring  $\mathbb{C}[[e_1, e_2, \dots, e_n]]$ . In this case, the roots of  $F(y, e_1, e_2, \dots, e_n)$  are  $x_1, x_2, \dots, x_n$ . We want to express roots of this  $F$  as power series in  $e_1, e_2, \dots, e_n$ . For this purpose, we consider a slightly modified version of  $F$ . More specifically, consider:

$$F(y, e_1, e_2, \dots, e_n) = y^n - e_1 y^{n-1} + \dots + (-1)^n (e_n + (-1)^{n-1}). \quad (6)$$

Notice that  $F(y, 0, 0, \dots, 0)$  has  $n$  distinct roots, namely the  $n^{\text{th}}$  roots of unity. Thus the roots of this  $F(y)$  (Equation 6) can be expressed as power series in  $e_1, e_2, \dots, e_n$ , this follows from Lemma 14. Let us record this as corollary 15.

► **Corollary 15.** *If  $F$  is as in Equation 6, then there exist  $n$  power series  $A_1, A_2, \dots, A_n \in \mathbb{C}[[e_1, e_2, \dots, e_n]]$  such that  $F(A_i) = 0$  for all  $i \in [n]$ .*

Now we show how to compute the degree  $d$  truncations of such roots  $A_1, A_2, \dots, A_n$ . This already follows from [9]. For the reader's convenience, we describe the algorithm here and a proof of correctness can be found in Appendix A.

## 4.2 Newton's Method

For the analysis of Algorithm 1, define the ideal  $I$  as:

$$I \stackrel{\text{def}}{=} (u_1, u_2, \dots, u_n).$$

► **Theorem 16.** *In Algorithm 1,  $A_i^{(k)} \equiv A_i \pmod{I^{2^k}}$  for all  $0 \leq k \leq \ell$  and for all  $i \in [n]$ .*

In Algorithm 1, we need to compute the inverse of  $F'(A^{(k)})$ , since we want to compute  $A^{(k+1)}$ , it is enough to compute the inverse of  $F'(A^{(k)}) \pmod{I^{2^{k+1}}}$ . This also follows from [9]. We explicitly describe this in Algorithm 2.

► **Lemma 17.** *Algorithm 2 computes a polynomial  $p$  such that  $p \equiv g^{-1} \pmod{I^d}$ .*

**Algorithm 2** Inverse computation.

**Input:** A circuit  $C$  computing the polynomial  $g(u_1, u_2, \dots, u_n)$  such that  $g(0, 0, \dots, 0) \neq 0$  and a positive integer  $d$  with  $d = 2^\ell$  for some  $\ell \in \mathbb{N}$ .

**Output:** A circuit  $D$  for computing a polynomial  $p(u_1, u_2, \dots, u_n)$  such that  $p \equiv g^{-1} \pmod{I^d}$ , here  $I = (u_1, u_2, \dots, u_n)$  and  $g^{-1}$  is the inverse of  $g$  in  $\mathbb{C}[[u_1, u_2, \dots, u_n]]$ .

```

1:  $p_0 \leftarrow \frac{1}{g(0,0,\dots,0)}$ .
2: for  $0 \leq k \leq \ell - 1$  do
3:    $p_{k+1} \leftarrow p_k \cdot (2 - g \cdot p_k)$ .
4: end for
5: return  $p_\ell$ .

```

We can also prove that there is a “small” circuit for  $p$  in Lemma 17.

► **Lemma 18.** *Let  $g(u_1, u_2, \dots, u_n)$  be a polynomial such that  $g(0, 0, \dots, 0) \neq 0$ . For any positive integer  $d$  with  $d = 2^\ell$  for some  $\ell \in \mathbb{N}$ , there is a polynomial  $p \in \mathbb{C}[u_1, u_2, \dots, u_n]$  such that  $p \equiv g^{-1} \pmod{I^d}$ . Moreover,  $L(p) \leq L(g) + O(\ell)$ .*

**Proof.** In Algorithm 2, we need three arithmetic operations to compute  $p_{k+1}$  from  $p_k$ . It follows from Lemma 17 that  $p_\ell = g^{-1} \pmod{I^d}$ . Thus there exists a circuit of size  $L(g) + 3 \cdot \ell = L(g) + O(\ell)$  computing  $p \equiv g^{-1} \pmod{I^d}$ . ◀

Now the following Theorem 19 follows by applying Lemma 18 and Theorem 16.

► **Theorem 19.** *Let  $F(y, e_1, e_2, \dots, e_n) = y^n - e_1 y^{n-1} + \dots + (-1)^n (e_n + (-1)^{n-1})$  and let  $A_1, A_2, \dots, A_n \in \mathbb{C}[[e_1, e_2, \dots, e_n]]$  such that  $F(A_i, e_1, e_2, \dots, e_n) = 0$  for all  $i \in [n]$ . Let  $d$  be a positive integer with  $d = 2^\ell$  for some  $\ell \in \mathbb{N}$  and let  $I = (e_1, e_2, \dots, e_n)$  be the ideal generated by  $e_1, e_2, \dots, e_n$  in the polynomial ring  $\mathbb{C}[e_1, e_2, \dots, e_n]$ . Let polynomials  $D_i$  be such that  $D_i \equiv A_i \pmod{I^d}$ . Then  $L(\{D_1, D_2, \dots, D_n\}) \leq O(n^2 \ell + n \ell^2)$ .*

**Proof.** We construct a circuit  $D$  whose outputs are  $D_1, D_2, \dots, D_n$ . We construct the desired circuit  $D$  by using Algorithm 1 on  $F(y, e_1, e_2, \dots, e_n)$  and  $s = (0, 0, \dots, 0)$ . It is enough to describe a circuit computing each  $D_i$  such that  $D_i \equiv A_i \pmod{I^d}$ . The circuit for  $A_i^{(0)}$  in Algorithm 1 is trivially of size one. By Step 5 of Algorithm 1, a circuit for  $A_i^{(k+1)}$  can be constructed given any circuits for  $A_i^{(k)}$ ,  $F(A_i^{(k)})$  and  $F'(A_i^{(k)})$ . Note that there are circuits of size  $O(n)$  computing  $F(y, e_1, e_2, \dots, e_n)$  and  $F'(y, e_1, e_2, \dots, e_n) \stackrel{\text{def}}{=} \frac{\partial F(y, e_1, e_2, \dots, e_n)}{\partial y}$ . Thus if  $A_i^{(k)}$  has a circuit of size  $s$ , then there exists a size  $s + O(n + \log d)$  circuit computing  $A_i^{(k+1)}$ , this follows from Lemma 18. In particular, there exists a circuit computing  $D_i \stackrel{\text{def}}{=} A_i^{(\lceil \log d \rceil)}$  of size  $O(n \log d + \log^2 d)$ . By Theorem 16, it follows that  $D_i \equiv A_i \pmod{I^d}$ . We combine the circuits computing  $D_i$ 's to construct the desired circuit  $D$  of size  $O(n \cdot n \log d + n \log^2 d) = O(n^2 \log d + n \log^2 d)$ . ◀

Now we are ready to prove Theorem 4.

**Proof of Theorem 4.** The main idea is what we have hinted above. Namely, let  $F(y, e_1, e_2, \dots, e_n)$  be the following polynomial:

$$F(y, e_1, e_2, \dots, e_n) = y^n - e_1 y^{n-1} + \dots + (-1)^n (e_n). \quad (7)$$

Here  $e_i = e_i^n$  is the  $i^{\text{th}}$  elementary symmetric polynomial. We know that the roots (as a uni-variate polynomial in  $y$ ) of  $F$  are  $x_1, x_2, \dots, x_n$ . Therefore,  $x_1, x_2, \dots, x_n$  are algebraic functions in  $e_1, e_2, \dots, e_n$ . Thus  $x_i = A_i(e_1, e_2, \dots, e_n)$  for some algebraic function  $A_i$ .

## 47:10 Complexity of Symmetric Polynomials

Let  $C_{\text{Sym}}(x_1, x_2, \dots, x_n)$  be a circuit of size  $L(f_{\text{Sym}})$  computing  $f_{\text{Sym}}(x_1, x_2, \dots, x_n)$ . If we could substitute the  $x_i$ 's by the  $A_i$ 's in  $C_{\text{Sym}}(x_1, x_2, \dots, x_n)$ , we would obtain a circuit for  $f$ . But we cannot compute algebraic functions using arithmetic circuits. Now replace  $e_n$  by  $e_n + (-1)^{n-1}$  in Equation 7. Thus the new  $F(y, e_1, e_2, \dots, e_n)$  is:

$$F(y, e_1, e_2, \dots, e_n) = y^n - e_1 y^{n-1} + \dots + (-1)^n (e_n + (-1)^{n-1}). \quad (8)$$

We call the roots of  $F(y)$  in Equation 8 again  $A_1, A_2, \dots, A_n$ . By using Lemma 14, we know that the  $A_i$ 's are in  $\mathbb{C}[[e_1, e_2, \dots, e_n]]$ . The following Equation 9 follows from the above discussion:

$$C_{\text{Sym}}(A_1, A_2, \dots, A_n) = f(e_1, e_2, \dots, e_n + (-1)^{n-1}). \quad (9)$$

To compute  $f$ , it is enough to substitute the degree  $d$  truncations of the  $A_i$ 's in Equation 9, instead of the exact infinite power series  $A_i$ . Let  $D_1, D_2, \dots, D_n$  be the outputs of the circuit  $D$  obtained by applying Theorem 19 with degree  $2^{\lceil \log d \rceil}$ . We substitute the  $x_i \rightarrow D_i$  in the circuit  $C_{\text{Sym}}(x_1, x_2, \dots, x_n)$ . We obtain the following equality:

$$h \stackrel{\text{def}}{=} C_{\text{Sym}}(D_1, D_2, \dots, D_n) = f(e_1, e_2, \dots, e_n + (-1)^{n-1}) + g. \quad (10)$$

In the above Equation 10,  $g$  is a polynomial with all its monomials of degree at least  $d+1$ , i.e.,  $g \in I^{d+1}$  with  $I = (e_1, e_2, \dots, e_n)$ . Hence it follows that:

$$f(e_1, e_2, \dots, e_n + (-1)^{n-1}) = \sum_{i=0}^d h^{[i]}.$$

By applying Theorem 19, we know that  $L^{f_{\text{Sym}}}(h) \leq (n^2 \log d + n \log^2 d)$ . Note that the degree of each  $D_i$  is at most  $2^{\lceil \log d \rceil}$ , which is at most  $2d$ . Thus the degree of  $h$  is at most  $2dd_{\text{Sym}}$ . By using Lemma 12, we conclude that for any  $0 \leq i \leq \deg(h)$ :

$$L^h(h^{[i]}) \leq O(n \cdot d \cdot d_{\text{Sym}}). \quad (11)$$

Equation 11 implies that  $L^h(\sum_{i=0}^d h^{[i]}) \leq O(n \cdot d^2 \cdot d_{\text{Sym}})$ . By using  $L^{f_{\text{Sym}}}(h) \leq (n^2 \log d + n \log^2 d)$ , we obtain that:

$$\begin{aligned} L^{f_{\text{Sym}}}\left(\sum_{i=0}^d h^{[i]}\right) &\leq O(n \cdot d^2 \cdot d_{\text{Sym}} \cdot (n^2 \log d + n \log^2 d)) \\ &= \tilde{O}(n^3 \cdot d^2 \cdot d_{\text{Sym}}). \end{aligned}$$

By using the substitution  $e_n \rightarrow e_n - (-1)^{n-1}$ , we obtain that:

$$L^{f_{\text{Sym}}}(f) \leq \tilde{O}(n^3 \cdot d^2 \cdot d_{\text{Sym}}).$$

If we use Lemma 13 instead of Lemma 12 in the above argument, we obtain that:

$$\begin{aligned} L(f) &\leq O(d^2(L(f_{\text{Sym}}) + n^2 \log d + n \log^2 d)) \\ &= \tilde{O}(d^2 L(f_{\text{Sym}}) + d^2 n^2). \end{aligned}$$

This concludes the proof. ◀

► **Remark.** In contrast to results in [7, 6], our results do depend on the degree  $d$ . But if the degree  $d$  is  $\text{poly}(n)$  then our upper bound on  $L(f)$  is polynomial in  $n$  and  $L(f_{\text{Sym}})$ . This upper bound was exponential in [7, 6].

## 5 Hard Symmetric Polynomials

By using Theorem 4, we are ready to prove that there exist *hard* symmetric polynomials. To this end, the following Theorem 20 suffices.

► **Theorem 20.** *Let  $(f_n)_{n \in \mathbb{N}}$  be a VNP-complete family. Then the corresponding symmetric polynomial family  $((f_n)_{\text{Sym}})_{n \in \mathbb{N}}$  is VNP-complete under  $c$ -reductions.*

**Proof.** Let  $d = \deg(f_n)$  and  $d_{\text{Sym}} = \deg((f_n)_{\text{Sym}})$ . Since  $(f_n)_{n \in \mathbb{N}} \in \text{VNP}$ , we know that both  $d$  and  $d_{\text{Sym}}$  are polynomially bounded in  $n$ . By using Theorem 4, we know that  $L^{f_{\text{Sym}}}(f) \leq \tilde{O}(n^3 \cdot d^2 \cdot d_{\text{Sym}}) = \tilde{O}(\text{poly}(n))$ . Thus  $((f_n)_{\text{Sym}})_{n \in \mathbb{N}}$  is VNP-hard under  $c$ -reductions. It is also easy to see that  $((f_n)_{\text{Sym}})_{n \in \mathbb{N}}$  is in VNP. Therefore  $((f_n)_{\text{Sym}})_{n \in \mathbb{N}}$  is VNP-complete under  $c$ -reductions. ◀

► **Corollary 21.** *The polynomial family  $(q_n)_{n \in \mathbb{N}}$  defined by  $q_n \stackrel{\text{def}}{=} (\text{per}_n)_{\text{Sym}}$  is VNP-complete under  $c$ -reductions. Therefore if  $\text{VP} \neq \text{VNP}$ , then the polynomial family  $(q_n)_{n \in \mathbb{N}}$  is not in VP.*

---

### References

- 1 E. Allender, P. Bürgisser, J. Kjeldgaard-Pedersen, and P. Miltersen. On the Complexity of Numerical Analysis. *SIAM Journal on Computing*, 38(5):1987–2006, 2009. doi:10.1137/070697926.
- 2 Garrett Birkhoff and Saunders Mac Lane. *A survey of modern algebra*. New York : Macmillan, 4th ed edition, 1977. URL: <http://www.gbv.de/dms/hbz/toc/ht000038471.pdf>.
- 3 Ben Blum-Smith and Samuel Coskey. The Fundamental Theorem on Symmetric Polynomials: History’s First Whiff of Galois Theory. *The College Mathematics Journal*, 48(1):18–29, 2017. URL: <http://www.jstor.org/stable/10.4169/college.math.j.48.1.18>.
- 4 J. N. Bray, M. D. E. Conder, C. R. Leedham-Green, and E. A. O’Brien. Short presentations for alternating and symmetric groups. *Trans. Amer. Math. Soc.*, 363(6):3277–3285, 2011. doi:10.1090/S0002-9947-2011-05231-1.
- 5 Peter Bürgisser. *Completeness and reduction in algebraic complexity theory*, volume 7. Springer Science & Business Media, 2013.
- 6 Xavier Dahan, Éric Schost, and Jie Wu. Evaluation properties of invariant polynomials. *Journal of Symbolic Computation*, 44(11):1592–1604, 2009. In Memoriam Karin Gatermann. doi:10.1016/j.jsc.2008.12.002.
- 7 Pierrick Gaudry, Eric Schost, and Nicolas M. Thiéry. Evaluation properties of symmetric polynomials. *International Journal of Algebra and Computation*, 16(3):505–523, 2006. doi:10.1142/S0218196706003128.
- 8 Oded Goldreich. *Computational complexity - a conceptual perspective*. Cambridge University Press, 2008.
- 9 H. T. Kung and J. F. Traub. All Algebraic Functions Can Be Computed Fast. *J. ACM*, 25(2):245–260, April 1978. doi:10.1145/322063.322068.
- 10 Dick Lipton and Ken Regan. Arithmetic Complexity and Symmetry, July 2009. URL: <https://rjlipton.wordpress.com/2009/07/10/arithmetic-complexity-and-symmetry/>.
- 11 Meena Mahajan. Algebraic Complexity Classes. In Manindra Agrawal and Vikraman Arvind, editors, *Perspectives in Computational Complexity: The Somenath Biswas Anniversary Volume*, pages 51–75. Springer International Publishing, Cham, 2014. doi:10.1007/978-3-319-05446-9\_4.
- 12 Tateaki Sasaki and Fujio Kako. Solving multivariate algebraic equation by Hensel construction. *Japan Journal of Industrial and Applied Mathematics*, 16(2):257–285, June 1999. doi:10.1007/BF03167329.

- 13 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. doi:10.1561/04000000039.
- 14 Volker Strassen. Vermeidung von Divisionen. *Journal für die reine und angewandte Mathematik*, 264:184–202, 1973.
- 15 L. G. Valiant. Completeness Classes in Algebra. In *Proceedings of the Eleventh Annual ACM Symposium on Theory of Computing*, STOC '79, pages 249–261, New York, NY, USA, 1979. ACM. doi:10.1145/800135.804419.

## A Appendix

### A.1 Algebraic complexity theory

Analogous to the idea of classical complexity classes, we can also define algebraic complexity classes. We refer the reader to [5, 11] for a more comprehensive introduction to algebraic complexity classes. In this section, a  $p$ -bounded function is simply a polynomially bounded function.

► **Definition 22** (Arithmetic Circuit Complexity). For a polynomial  $p \in \mathbb{F}[x_1, x_2, \dots, x_n]$ , the (arithmetic) circuit complexity  $L(p)$  of  $p$  is defined as the size of smallest arithmetic circuit computing  $p$ , that is

$$L(p) \stackrel{\text{def}}{=} \min\{s \mid \exists \text{ size } s \text{ arithmetic circuit computing } p\}.$$

► **Definition 23** ( $p$ -family). A family (or a sequence)  $(f_n)_{n \in \mathbb{N}}$  of (multivariate) polynomials over the field  $\mathbb{F}$  is said to be a  $p$ -family iff the number of variables as well as the degree of  $f_n$  are  $p$ -bounded functions of  $n$ .

Now we define the notion of *efficient* polynomial families.

► **Definition 24** ( $p$ -computable). A  $p$ -family  $(f_n)_{n \in \mathbb{N}}$  is called  $p$ -computable iff the arithmetic complexity  $L(f_n)$  is a  $p$ -bounded function of  $n$ .

$p$ -computable polynomial families define the algebraic analogue of the P, called VP.

► **Definition 25** (Class VP). The (algebraic complexity) class VP is the set of all  $p$ -computable polynomial families.

► **Definition 26** (Class VNP). A  $p$ -family  $(f_n)_{n \in \mathbb{N}}$  is said to be in the (algebraic complexity) class VNP if there exists a polynomial family  $(g_n)_{n \in \mathbb{N}} \in \text{VP}$  with  $g_n \in \mathbb{F}[x_1, x_2, \dots, x_{q(n)}]$  such that:

$$f_n(x_1, x_2, \dots, x_{p(n)}) = \sum_{e \in \{0,1\}^{q(n)-p(n)}} g_n(x_1, x_2, \dots, x_{p(n)}, e_1, e_2, \dots, e_{q(n)-p(n)}).$$

Similar to the Boolean case, there is an algebraic notion of reduction also, called the  $p$ -projections.

► **Definition 27** (Projection). A polynomial  $f(x_1, x_2, \dots, x_n) \in \mathbb{F}[x_1, x_2, \dots, x_n]$  is said to be a projection of a polynomial  $g(y_1, y_2, \dots, y_m) \in \mathbb{F}[y_1, y_2, \dots, y_m]$ , if there exists a map  $\alpha : \{y_1, y_2, \dots, y_m\} \rightarrow \{x_1, x_2, \dots, x_n\} \cup \mathbb{F}$  such that  $f = g$  under the substitution map  $\alpha$ . We write  $f \leq g$  to denote that  $f$  is a projection of  $g$ .

► **Definition 28** (*p*-projection). A polynomial family  $(f_n)_{n \in \mathbb{N}}$  is said to be a *p*-projection of a polynomial family  $(g)_{n \in \mathbb{N}}$  if there is a *p*-bounded function  $\beta : \mathbb{N} \rightarrow \mathbb{N}$  and  $n_0 \in \mathbb{N}$  such that:

$$\forall n \geq n_0 : f_n \leq g_{\beta(n)}.$$

We denote  $(f_n)_{n \in \mathbb{N}}$  being a *p*-projection of  $(g)_{n \in \mathbb{N}}$  by  $f \leq_p g$ .

Definition 9 naturally lends to the following definition:

► **Definition 29** ([5]). Let  $f = (f_n), g = (g_n)$  be two polynomial families. We say that  $f$  is a *c*-reduction of  $g$ , denoted by  $f \leq_c g$ , iff there is a *p*-bounded function  $t : \mathbb{N} \rightarrow \mathbb{N}$  such that  $L^{g^{t(n)}}(f_n)$  is a *p*-bounded function of  $n$ .

Now the idea of completeness and hardness can be defined as in the case of Boolean case.

► **Definition 30** (Hardness and Completeness). For an algebraic complexity classic  $\mathcal{C}$ , a *p*-family  $f = (f_n)_{n \in \mathbb{N}}$  is said to be a  $\mathcal{C}$ -hard if  $g \leq_p f$  for all  $g \in \mathcal{C}$ ,  $f$  is called  $\mathcal{C}$ -complete if  $f$  is  $\mathcal{C}$ -hard and  $f \in \mathcal{C}$ . Similarly, we can define the notion of hardness under *c*-reductions.

► **Theorem 31** ([15], see also [5]). *Over the fields  $\mathbb{F}$  with  $\text{char}(\mathbb{F}) \neq 2$ , the *p*-family  $(\text{per}_n)$  is VNP-complete.*

The holy grail of algebraic complexity theory is to show that  $\text{VP} \neq \text{VNP}$ . For this it is enough to show that  $(\text{per}_n) \notin \text{VP}$  over fields  $\mathbb{F}$  with  $\text{char}(\mathbb{F}) \neq 2$ . Note that  $\text{per}_n$  and  $\text{det}_n$  are the same polynomials if  $\text{char}(\mathbb{F}) = 2$ . Thus if  $\text{char}(\mathbb{F}) = 2$  then  $(\text{per}_n) \in \text{VP}$  hence  $(\text{per}_n)$  is unlikely to VNP-complete over fields of characteristic two.

## A.2 Missing proofs

We provide some proofs of known results used in this work for the reader's convenience.

**Proof of Lemma 13.** Let  $C$  be a circuit of size  $L(f)$  computing  $f$ . We create  $m + 1$  copies of each arithmetic gate in  $C$ , i.e., each  $\{+, -, \times\}$ -gate  $G$  has  $m + 1$  copies  $G_0, G_1, \dots, G_m$ . If the gate  $G$  computes the polynomial  $g$  then  $G_i$  computes the polynomial  $g^{[i]}$ . This can be trivially done for input and constant gates. Suppose  $G = G_1 + G_2$  is a “+” gate and  $g_1, g_2$  are the polynomials computed by gates  $G_1$  and  $G_2$  respectively. Now we know that  $g^{[i]} = g_1^{[i]} + g_2^{[i]}$  for all  $i \in [[m]]$ . A similar statement is true for “−” gates also. If  $G = G_1 \times G_2$  is a “×” gate, then we have the following equality:

$$g^{[i]} = \sum_{j=0}^i g_1^{[j]} \cdot g_2^{[i-j]}. \tag{12}$$

Suppose we already have the gates for  $g_1^{[j]}, g_2^{[j]}$  for all  $j \in [[m]]$ . Then one  $g^{[i]}$  in Equation 12 can be computed using  $2(i + 1)$  additional gates. Thus the gates  $G_0, G_1, \dots, G_m$  can be constructed using  $\sum_{k=0}^m 2(k + 1) = O(m^2)$  gates. Hence every gate in  $C$  corresponds to at most  $O(m^2)$  new gates. Therefore:

$$L(\{f^{[0]}, f^{[1]}, \dots, f^{[m]}\}) \leq O(m^2 L(f)). \quad \blacktriangleleft$$

**Proof of Theorem 16.** Our claim is obviously true for  $k = 0$ . We prove the theorem by induction on  $k$ . We prove it simultaneously for all  $i \in [n]$ , so for the sake of brevity we use  $A^{(k)}$  to denote  $A_i^{(k)}$  and  $\alpha$  to denote  $\alpha_i$ . Consider the following equalities for a root  $A = A_i$  of  $F(y)$ :

$$\begin{aligned} 0 &= F(A) = F(A^{(k)} + (A - A^{(k)})) \\ &= F(A^{(k)}) + (A - A^{(k)})F'(A^{(k)}) + \sum_{j>1} \frac{(A - A^{(k)})^j}{j!} F^{(j)}(A^{(k)}). \end{aligned} \tag{13}$$

## 47:14 Complexity of Symmetric Polynomials

Here  $F^{(j)} \stackrel{\text{def}}{=} \frac{\partial^j F(y, u_1, u_2, \dots, u_n)}{\partial y^j}$  is the  $j^{\text{th}}$  derivative of  $F(y)$  with respect to  $y$ . Since  $\alpha$  is a simple root of  $F(y, 0, 0, \dots, 0)$ , we know that:

$$\text{constant term of } F'(A^{(k)}) = F'(\alpha) \neq 0.$$

Thus  $F'(A_k)$  is invertible in the ring  $\mathbb{C}[[u_1, u_2, \dots, u_n]]$  of power series. Therefore we have:

$$\begin{aligned} A - A^{(k+1)} &= A - \left( A^{(k)} - \frac{F(A^{(k)})}{F'(A^{(k)})} \right) \\ &= - \sum_{j>1} \frac{(A - A^{(k)})^j F^{(j)}(A^{(k)})}{j! \cdot F'(A^{(k)})}. \end{aligned} \quad (\text{by using Equation 13})$$

Since  $A - A^{(k)} \in I^{2^k}$ , the right hand side of the above equation is in  $I^{2^{k+1}}$ . Thus  $A^{(k+1)} \equiv A \pmod{I^{2^{k+1}}}$ . ◀

**Proof of Lemma 17.** We again prove it by induction on  $k$ , the induction hypothesis is that  $p_k \equiv g^{-1} \pmod{I^{2^k}}$ . This induction hypothesis is trivially true for  $k = 0$ . Consider:

$$\begin{aligned} \frac{1}{g} - p_{k+1} &= \frac{1}{g} - p_k(2 - g \cdot p_k) \\ &= g \cdot \left( \frac{1}{g^2} - \frac{2p_k}{g} + p_k^2 \right) \\ &= g \cdot \left( \frac{1}{g} - p_k \right)^2. \end{aligned}$$

By using the induction hypothesis, we know that  $\frac{1}{g} - p_k \in I^{2^k}$ . Therefore it implies that  $\frac{1}{g} - p_{k+1} \in I^{2^{k+1}}$ . Now the lemma follows from the fact that  $\ell = \lceil \log d \rceil$ . ◀




# The Orthogonal Vectors Conjecture for Branching Programs and Formulas

Daniel M. Kane<sup>1</sup>

CSE and Mathematics, UC San Diego, La Jolla CA, USA  
dakane@ucsd.edu

Richard Ryan Williams<sup>2</sup>

EECS and CSAIL, MIT, 32 Vassar St., Cambridge MA, USA  
rrw@mit.edu

 <https://orcid.org/0000-0003-2326-2233>

---

## Abstract

---

In the **ORTHOGONAL VECTORS (OV)** problem, we wish to determine if there is an orthogonal pair of vectors among  $n$  Boolean vectors in  $d$  dimensions. The *OV Conjecture (OVC)* posits that **OV** requires  $n^{2-o(1)}$  time to solve, for all  $d = \omega(\log n)$ . Assuming the OVC, optimal time lower bounds have been proved for many prominent problems in **P**, such as Edit Distance, Frechet Distance, Longest Common Subsequence, and approximating the diameter of a graph.

We prove that OVC is true in several computational models of interest:

- For all sufficiently large  $n$  and  $d$ , **OV** for  $n$  vectors in  $\{0, 1\}^d$  has branching program complexity  $\tilde{\Theta}(n \cdot \min(n, 2^d))$ . In particular, the lower and upper bounds match up to polylog factors.
- **OV** has Boolean formula complexity  $\tilde{\Theta}(n \cdot \min(n, 2^d))$ , over all complete bases of  $O(1)$  fan-in.
- **OV** requires  $\tilde{\Theta}(n \cdot \min(n, 2^d))$  wires, in formulas comprised of gates computing arbitrary symmetric functions of unbounded fan-in.

Our lower bounds basically match the best known (quadratic) lower bounds for *any* explicit function in those models. Analogous lower bounds hold for many related problems shown to be hard under OVC, such as Batch Partial Match, Batch Subset Queries, and Batch Hamming Nearest Neighbors, all of which have very succinct reductions to **OV**.

The proofs use a certain kind of input restriction that is different from typical random restrictions where variables are assigned independently. We give a sense in which independent random restrictions cannot be used to show hardness, in that OVC is false in the “average case” even for  $AC^0$  formulas:

For all  $p \in (0, 1)$  there is a  $\delta_p > 0$  such that for every  $n$  and  $d$ , **OV** instances with input bits independently set to 1 with probability  $p$  (and 0 otherwise) can be solved with  $AC^0$  formulas of  $O(n^{2-\delta_p})$  size, on all but a  $o_n(1)$  fraction of instances. Moreover,  $\lim_{p \rightarrow 1} \delta_p = 1$ .

**2012 ACM Subject Classification** Theory of computation  $\rightarrow$  Circuit complexity

**Keywords and phrases** fine-grained complexity, orthogonal vectors, branching programs, symmetric functions, Boolean formulas

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2019.48

**Acknowledgements** We are very grateful to Ramamohan Paturi for raising the question of whether the OV conjecture is true for  $AC^0$  circuits. We also thank Abhishek Bhrushundi, Ludmila Glinskikh, and the anonymous reviewers for helpful comments.

---

<sup>1</sup> Supported by NSF Award CCF-1553288 (CAREER) and a Sloan Research Fellowship.

<sup>2</sup> Supported by NSF CCF-1741615 (CAREER: Common Links in Algorithms and Complexity). Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.



## 1 Introduction

We investigate the following basic problem:

**ORTHOGONAL VECTORS (OV)**  
 Given:  $n$  vectors  $v_1, \dots, v_n \in \{0, 1\}^d$   
 Decide: Are there  $i, j$  such that  $\langle v_i, v_j \rangle = 0$ ?

An instructive way of viewing the **OV** problem is that we have a collection of  $n$  sets over  $[d]$ , and wish to find two disjoint sets among them. The obvious algorithm runs in time  $O(n^2 \cdot d)$ , and  $\log(n)$  factors can be shaved [33]. For  $d < \log_2(n)$ , stronger improvements are possible: there are folklore  $O(n \cdot 2^d \cdot d)$ -time and  $\tilde{O}(n + 2^d)$ -time algorithms (for a reference, see [1]). Truly subquadratic-time algorithms have recently been developed for even larger dimensionalities: the best known result in this direction is that for all constants  $c \geq 1$ , **OV** with  $d = c \log n$  dimensions can be solved in  $n^{2-1/O(\log c)}$  time [7, 19]. However, it seems inherent that, as the vector dimension  $d$  increases significantly beyond  $\log n$ , the time complexity of **OV** approaches the trivial  $n^2$  bound.

Over the last several years, a significant body of work has been devoted to understanding the following plausible lower bound conjecture:

► **Conjecture 1** (Orthogonal Vectors Conjecture (OVC) [37, 6, 9, 3]). *For every  $\varepsilon > 0$ , there is a  $c \geq 1$  such that **OV** cannot be solved in  $n^{2-\varepsilon}$  time on instances with  $d = c \log n$ .*

In other words, OVC states that **OV** requires  $n^{2-o(1)}$  time on instances of dimension  $\omega(\log n)$ . The popular Strong Exponential Time Hypothesis [27, 18] (on the time complexity of CNF-SAT) implies OVC [37]. For this reason, and the fact that the **OV** problem is very simple to work with, the OVC has been the engine under the hood of many recent conditional lower bounds on classic problems solvable within P. For example, the OVC implies nearly-quadratic time lower bounds for Edit Distance [9], approximating the diameter of a graph [34], Frechet Distance [13, 15], Longest Common Substring and Local Alignment [6], Regular Expression Matching [10], Longest Common Subsequence, Dynamic Time Warping, and other string similarity measures [3, 14], Subtree Isomorphism and Largest Common Subtree [2], Curve Simplification [16], intersection emptiness of two finite automata [35], first-order properties on sparse finite structures [24] as well as average-case hardness for quadratic-time [11]. Other works surrounding the OVC (or assuming it) include [38, 36, 5, 20, 8, 23, 26, 17, 30, 22].

Therefore it is of strong interest to prove the OVC in reasonable computational models. Note that **OV** can be naturally expressed as a depth-three formula with unbounded fan-in: an OR of  $n^2$  NORs of  $d$  ANDs on two input variables: an  $AC^0$  formula of size  $O(n^2 \cdot d)$ . Are there smaller formulas for **OV**?

### 1.1 OVC is True in Restricted Models

In this paper, we study how well **OV** can be solved in the Boolean formula and branching program models. Among the aforementioned **OV** algorithms, only the first two seem to be efficiently implementable by formulas and branching programs: for example, there are DeMorgan formulas for **OV** of size only  $O(n^2 d)$  and size  $O(nd2^d)$ , respectively (see Proposition 4).

The other algorithms do not seem to be implementable in small space, in particular with small-size branching programs. Our first theorem shows that the simple constructions solving **OV** with  $O(n^2 \cdot d)$  and  $O(n \cdot 2^d \cdot d)$  work are essentially optimal for all choices of  $d$  and  $n$ :

► **Theorem 2** (OVC For Formulas of Bounded Fan-in). *For every constant  $c \geq 1$ , **OV** on  $n$  vectors in  $d$  dimensions does not have  $c$ -fan-in formulas of size  $O(\min\{n^2/(\log d), n \cdot 2^d/(d^{1/2} \log d)\})$ , for all sufficiently large  $n, d$ .*

► **Theorem 3** (OVC For Branching Programs). ***OV** on  $n$  vectors in  $d$  dimensions does not have branching programs of size  $O(\min\{n^2, n \cdot 2^d/(d^{1/2})\}/(\log(nd) \log(d)))$ , for all sufficiently large  $n, d$ .*

As far as we know, size- $s$  formulas of constant fan-in may be more powerful than size- $s$  branching programs (but note that DeMorgan formulas can be efficiently simulated with branching programs). Thus the two lower bounds are incomparable. These lower bounds are tight up to the (negligible) factor of  $\min\{\sqrt{\log n}, d^{1/2}\} \log(d) \log(nd)$ , as the following simple construction shows:

► **Proposition 4.** ***OV** has  $AC^0$  formulas (and branching programs) of size  $O(dn \cdot \min(n, 2^d))$ .*

**Proof.** The  $O(dn^2)$  bound is obvious: take an OR over all  $\binom{n}{2}$  pairs of vectors, and use an  $AND \circ OR$  of  $O(d)$  size to determine orthogonality of the pair. For the  $O(dn2^d)$  bound, our strategy is to try all  $2^d$  vectors  $v$ , and look for a  $v$  that is equal to one input vector and is orthogonal to another input vector. To this end, take an OR over all  $2^d$  possible vectors  $w$  over  $[d]$ , and take the AND of two conditions:

1. There is a vector  $v$  in the input such that  $v = w$ . This can be computed with an OR over all  $n$  vectors of an  $O(d)$ -size formula, in  $O(nd)$  size.
2. There is a vector  $u$  in the input such that  $\langle u, w \rangle = 0$ . This can be computed with a *parallel* OR over all  $n$  vectors of an  $O(d)$ -size formula, in  $O(nd)$  size.

Note that the above formulas have constant-depth, with unbounded fan-in AND and OR gates. Since DeMorgan formulas of size  $s$  can be simulated by branching programs of size  $O(s)$ , the proof is complete.<sup>3</sup> ◀

### Formulas with symmetric gates

As mentioned above, **OV** can be naturally expressed as a depth-three formula of unbounded fan-in: an  $AC_3^0$  formula of  $O(n^2d)$  wires. We show that this wire bound is also nearly optimal, even when we allow arbitrary symmetric Boolean functions as gates. Note this circuit model subsumes both  $AC$  (made up of AND, OR, and NOT gates) and  $TC$  (made up of MAJORITY and NOT gates).

► **Theorem 5** (OVC For TC Formulas). *Every formula computing **OV** composed of arbitrary symmetric functions with unbounded fan-in needs at least  $\Omega(\min\{n^2/(\log d), n \cdot 2^d/(d^{1/2} \log d)\})$  wires, for all  $n$  and  $d$ .*

## 1.2 Lower Bounds for Batch Partial Match, Batch Subset Query, Batch Hamming Nearest Neighbors, etc.

A primary reason for studying **OV** is its ubiquity as a “bottleneck” special case of many other basic search problems. In particular, many problems have very succinct reductions from **OV** to them, and our lower bounds extend to these problems.

<sup>3</sup> This should be folklore, but we couldn’t find a reference; see the Appendix B. An anonymous referee thought we should attribute Borodin [12], but that reference only shows that circuits of depth  $s$  can be simulated in space  $O(s)$ . We need something much stronger: branching programs of size  $O(s)$  correspond to  $\log_2(s) + O(1)$  space. Lynch [31] showed that formulas of size  $s$  can be simulated in  $O(\log s)$  space, but in our case we need  $\log_2(s) + O(1)$  space.

We say that a *linear projection reduction* from a problem  $A$  to problem  $B$  is a circuit family  $\{C_n\}$  where each  $C_n$  has  $n$  input and  $O(n)$  outputs, each output of  $C_n$  depends on at most one input, and  $x \in A$  if and only if  $C_{|x|}(x) \in B$ , for all possible inputs  $x$ . Under this constrained reduction notion, it is easy to see that if **OV** has a linear projection reduction to  $B$ , then size lower bounds for **OV** (even in our restricted settings) imply analogous lower bounds for  $B$  as well. Via simple linear projection reductions which preserve both  $n$  and  $d$  (up to constant multiplicative factors), analogous lower bounds hold for many other problems which have been commonly studied, such as:

BATCH PARTIAL MATCH

Given:  $n$  “database” vectors  $v_1, \dots, v_n \in \{0, 1\}^d$  and  $n$  queries  $q_1, \dots, q_n \in \{0, 1, \star\}^d$   
Decide: Are there  $i, j$  such that  $v_i$  is a partial match of  $q_j$ , i.e. for all  $k$ ,  $q_j[k] \in \{v_i[k], \star\}$ ?

BATCH SUBSET QUERY

Given:  $n$  sets  $S_1, \dots, S_n \subseteq [d]$  and  $n$  queries  $T_1, \dots, T_n \subseteq [d]$   
Decide: Are there  $i, j$  such that  $S_i \subseteq T_j$ ?

BATCH HAMMING NEAREST NEIGHBORS

Given:  $n$  points  $p_1, \dots, p_n \in \{0, 1\}^d$  and  $n$  queries  $q_1, \dots, q_n \in \{0, 1\}^d$ , integer  $k$   
Decide: Are there  $i, j$  such that  $p_i$  and  $q_j$  differ in at most  $k$  positions?

### 1.3 “Average-Case” OVC is False, Even for AC<sup>0</sup>

The method of proof in the above lower bounds is an input restriction method that does *not* assign variables independently (to 0, 1, or  $\star$ ) at random. (Our restriction method could be viewed as a random process, just not one that assigns variables independently.) Does **OV** become easier under natural product distributions of instances, e.g., with each bit of each vector being an independent random variable? Somewhat surprisingly, we show that a reasonable parameterization of average-case OVC is false, even for AC<sup>0</sup> formulas.

For  $p \in (0, 1)$ , and for a given  $n$  and  $d$ , we call  $\mathbf{OV}(p)_{n,d}$  the distribution of **OV** instances where all bits of the  $n$  vectors are chosen independently, set to 1 with probability  $p$  and 0 otherwise. We would like to understand when  $\mathbf{OV}(p)_{n,d}$  can be efficiently solved on almost all instances (i.e., with probability  $1 - o(1)$ ). We give formulas of truly sub-quadratic size for every  $p > 0$ :

► **Theorem 6.** *For every  $p \in (0, 1)$ , and every  $n$  and  $d$ , there is an AC<sup>0</sup> formula of size  $n^{2-\varepsilon_p}$  that correctly answers all but a  $o_n(1)$  fraction of  $\mathbf{OV}(p)_{n,d}$  instances on  $n$  vectors and  $d$  dimensions, for an  $\varepsilon_p > 0$  such that  $\varepsilon_p \rightarrow 1$  as  $p \rightarrow 1$ .*

Interestingly, our AC<sup>0</sup> formulas have one-sided error, even in the worst case: if there is no orthogonal pair in the instance, our formulas *always* output 0. However, they may falsely report that there is no orthogonal pair, but this only occurs with probability  $o(1)$  on a random  $\mathbf{OV}(p)_{n,d}$  instance, for any  $n$  and  $d$ .

## 1.4 Intuition

Our lower bounds give some insight into why **OV** is hard. There are two main ideas:

1. **OV** instances with  $n$   $d$ -dimensional vectors can encode difficult Boolean functions on  $d$  inputs, requiring circuits of size  $\tilde{\Omega}(\min(2^d, n))$ . This can be accomplished by encoding those strings with “middle” Hamming weight from the truth table of a hard function with the vectors in an **OV** instance, in such a way that finding an orthogonal pair is equivalent to evaluating the hard Boolean function at a given  $d$ -bit input. This is an inherent property of **OV** that is independent of the computational model.
2. Because we are working with simple computational models, we can generally make the following kind of claim: given an algorithm for solving **OV** and given a partial assignment to all input vectors except for one appropriately chosen vector, we can propagate this partial assignment through the algorithm, and “shrink” the size of the algorithm by a factor of  $\Omega(n)$ . This sort of argument was first used by Nechiporuk [32] in the context of branching program lower bounds, and can be also applied to formulas.

Combining the two ideas, if we can “shrink” our algorithm by a factor of  $n$  by restricting the inputs appropriately, and argue that the remaining subfunction requires circuits of size  $\tilde{\Omega}(\min(2^d, n))$ , we can conclude that the original algorithm for **OV** must have had size  $\tilde{\Omega}(\min(n2^d, n^2))$ . (Of course, there are many details to verify, but this is the basic idea.)

The small  $\text{AC}^0$  formulas for **OV**( $p$ ) (the average-case setting) involve several ideas. First, given the probability  $p \in (0, 1)$  of 1 and the number of vectors  $n$ , we observe a simple phase transition phenomenon: there is only a particular range of dimensionality  $d$  in which the problem is non-trivial, and outside of this range, almost all instances are either “yes” instances or “no” instances. Second, within this “hard” range of  $d$ , the orthogonal vector pairs are expected to have a special property: with high probability, at least one orthogonal pair in a “yes” instance has noticeably fewer ones than a typical vector in the distribution. To obtain a sub-quadratic size  $\text{AC}^0$  formula from these observations, we partition the instance into small groups such that the orthogonal pair (if it exists) is the only “sparse” vector in its group, whp. Over all pairs of groups  $i, j$  in parallel, we take the component-wise OR of all sparse vectors in group  $i$  (call the resulting vector  $u_i$ ), and similarly for group  $j$  (call the result  $v_j$ ). Then we test if  $u_i$  and  $v_j$  are orthogonal. By doing so, if our formula ever reports 1, then there is some orthogonal pair in the instance (even in the worst case).

## 2 Lower Bounds

### Functions hard for the middle layer of the hypercube

In our lower bound proofs, we will use functions on  $d$ -inputs for which every small circuit fails to agree with the function on inputs of Hamming weight about  $d/2$ . Let  $\binom{[d]}{k}$  denote the set of all  $d$ -bit vectors of Hamming weight  $k$ .

► **Lemma 7.** *Let  $d$  be even, let  $\mathcal{C}$  be a set of Boolean functions, let  $N(d, s)$  be the number of functions in  $\mathcal{C}$  on  $d$  inputs of size at most  $s$ , and let  $s^* \in \mathbb{N}$  satisfy  $\log_2(N(d, s^*)) < \binom{[d]}{d/2}$ .*

*Then there is a sequence  $S$  of  $\binom{[d]}{d/2}$  pairs  $(x_i, y_i) \in \binom{[d]}{d/2} \times \{0, 1\}$ , such that each  $x_i \in \binom{[d]}{d/2}$  appears exactly once among the pairs in  $S$ , and every function  $f : \{0, 1\}^d \rightarrow \{0, 1\}$  satisfying  $f(x_i) = y_i$  (for all  $i = 1, \dots, \binom{[d]}{d/2}$ ) requires  $\mathcal{C}$ -size at least  $s^*$ .*

**Proof.** By definition, there are  $N(d, s)$  functions of size  $s$  on  $d$  inputs from  $\mathcal{C}$ , and there are  $2^{\binom{[d]}{d/2}}$  possible input/output sequences  $(x_i, y_i) \in \binom{[d]}{d/2} \times \{0, 1\}$  defined over all  $d$ -bit vectors of Hamming weight  $d/2$ . When  $2^{\binom{[d]}{d/2}} > N(d, s)$ , there is at least one input/output sequence that is not satisfied by any function in  $\mathcal{C}$  of size  $s$ . ◀

Note that it does not matter *what* is meant by “size” in the above lemma: the size measure could be gates, wires, etc., and the lemma still holds (as it is just counting). The above simple lemma applies to formulas, as follows:

► **Corollary 8.** *Let  $c \geq 2$  be a constant. There are  $\binom{d}{d/2}$  pairs  $(x_i, y_i) \in \binom{[d]}{d/2} \times \{0, 1\}$ , such that every function  $f : \{0, 1\}^d \rightarrow \{0, 1\}$  satisfying  $f(x_i) = y_i$  (for all  $i = 1, \dots, \binom{d}{d/2}$ ) needs  $c$ -fan-in formulas of size at least  $\Omega(2^d / (d^{1/2} \log d))$ .*

**Proof.** There are  $N(d, s) \leq d^{k_c \cdot s}$  formulas of size  $s$  on  $d$  inputs, where the constant  $k_c$  depends only on  $c$ . When  $2^{\binom{d}{d/2}} > d^{k_c \cdot s}$ , Lemma 7 says that there is an input/output sequence of length  $\binom{d}{d/2}$  that *no* formula of size  $s$  can satisfy. Thus to satisfy that sequence, we need a formula of size  $s$  at least large enough that  $2^{\binom{d}{d/2}} \leq d^{k_c \cdot s}$ , i.e.,  $s \geq \Omega\left(\binom{d}{d/2} / \log(d)\right) \geq \Omega(2^d / (d^{1/2} \log d))$ . ◀

## 2.1 Lower Bound for Constant Fan-in Formulas

We are now ready to prove the lower bound for Boolean formulas of constant fan-in:

► **Reminder of Theorem 2.** *For every constant  $c \geq 1$ , **OV** on  $n$  vectors in  $d$  dimensions does not have  $c$ -fan-in formulas of size  $O(\min\{n^2 / (\log d), n \cdot 2^d / (d^{1/2} \log d)\})$ , for all sufficiently large  $n, d$ .*

All of the lower bound proofs have a similar structure. We will give considerably more detail in the proof of Theorem 2 to aid the exposition of the later lower bounds.

**Proof.** To simplify the calculations, assume  $d$  is even in the following. Let  $F_{n,d+1}(v_1, \dots, v_n)$  be a  $c$ -fan-in formula of minimal size  $s$  computing **OV** on  $n$  vectors of dimension  $d + 1$ , where each  $v_i$  denotes a sequence of  $d + 1$  Boolean variables  $(v_{i,1}, \dots, v_{i,d+1})$ .

Let  $\ell$  be the number of leaves of  $F_{n,d+1}$ . Since  $F_{n,d+1}$  is minimal, each gate has fan-in at least two (gates of fan-in 1 can be “merged” into adjacent gates). Therefore (by an easy induction on  $s$ ) we have

$$s \geq \ell \geq s/2. \tag{1}$$

Observe there must be a vector  $v_{i^*}$  (for some  $i^* \in [n]$ ) whose  $d + 1$  Boolean variables appear on at most  $\ell/n$  leaves of the formula  $F_{n,d}$ . Our plan now is to leave the  $d + 1$  variables corresponding to  $v_{i^*}$  free, and set all other variables in a particular way, so that the remaining subfunction requires many gates. We consider two cases (based on how  $d$  compares with  $n$ ), which determine how we set those variables.

**Case 1.** Suppose  $\binom{d}{d/2} \leq n - 1$ . Let  $\{(x_i, y_i)\} \subseteq \binom{[d]}{d/2} \times \{0, 1\}$  be a list of hard pairs from Corollary 8, and let  $f : \{0, 1\}^d \rightarrow \{0, 1\}$  be any function that satisfies  $f(x_i) = y_i$ , for all  $i$ . By Corollary 8, such an  $f$  needs  $c$ -fan-in formulas of size at least  $\Omega(2^d / (d^{1/2} \log d))$ . Let  $\{x'_1, \dots, x'_t\} \subseteq \binom{[d]}{d/2}$  be those  $d$ -bit strings of Hamming weight  $d/2$  such that  $f(x'_i) = 1$ , for some  $t \leq \binom{d}{d/2} \leq n - 1$ .

**Case 2.** Suppose  $\binom{d}{d/2} \geq n - 1$ . Then we claim that there is a list of input/output pairs  $(x_1, y_1), \dots, (x_{n-1}, y_{n-1}) \in \binom{[d]}{d/2} \times \{0, 1\}$  such that for every  $f : \{0, 1\}^d \rightarrow \{0, 1\}$  satisfying  $f(x_i) = y_i$ , for all  $i$ ,  $f$  needs formulas of size at least  $\Omega(n / \log d)$ . To see this, note that if we take  $n - 1$  distinct strings  $x_1, \dots, x_{n-1}$  from  $\binom{[d]}{d/2}$ , there are  $2^{n-1}$  possible choices for the list of pairs. So when  $2^{n-1} > d^{k_c \cdot s}$ , there is a list of hard pairs  $(x_1, y_1), \dots,$



$(x_{n-1}, y_{n-1})$  that no formula of size  $s$  satisfies. For any function  $f : \{0, 1\}^d \rightarrow \{0, 1\}$  such that  $f(x_i) = y_i$  for all  $i = 1, \dots, n-1$ , its formula size  $s$  must be at least  $\Omega(n/\log d)$  in this case.

Let  $\{x'_1, \dots, x'_t\} \subseteq \binom{[d]}{d/2}$  be those  $d$ -bit strings of Hamming weight  $d/2$  such that  $(x'_i, 1)$  is on the list of hard pairs, for some  $t \leq n-1$ .

In either of the two cases, we will use the list of  $t \leq \min\{n-1, \binom{d}{d/2}\}$  strings  $\{x'_1, \dots, x'_t\}$  to assign all variables  $v_i$  of our **OV** formula, for all  $i \neq i^*$ . In particular, for the first  $t$  integers  $i \in [n]$  such that  $i \neq i^*$ , we substitute each  $(d+1)$ -bit input vector  $v_i$  with a distinct  $(d+1)$ -bit string  $\overline{1x'_{i'}}$  (the bit 1 concatenated with the complement of  $x_{i'}$ , obtained by flipping all the bits of  $x'_{i'}$ ). If  $t < n-1$  (which can happen in Case 1), we also substitute all other input vectors  $v_j$  where  $j \neq i^*$  with the  $(d+1)$ -bit vector  $\vec{1}$ . Note that all of the pairs of vectors substituted so far are not orthogonal to each other: for all  $i \neq i'$ , we have  $\langle \overline{1x'_i}, \overline{1x'_{i'}} \rangle \neq 0$ , and for all  $i$  we have  $\langle \overline{1x'_i}, \vec{1} \rangle \neq 0$ . Finally, we substitute a 0 in the first input bit of  $\vec{v}_{i^*}$ .

After these substitutions, the remaining formula  $F'_n$  has only  $d$  inputs, namely the last  $d$  bits of the vector  $v_{i^*}$ . Moreover,  $F'_n$  is a formula with at most  $\ell/n$  leaves labeled by literals: the rest of the leaves are labeled with 0/1 constants. After simplifying the formula (replacing all gates with some 0/1 inputs by equivalent functions of smaller fan-in, and replacing gates of fan-in 1 by wires), the total number of leaves of  $F'_n$  is now at most  $\ell/n$ . Therefore by (1) we infer that

$$\text{size}(F'_n) \leq 2\ell/n. \quad (2)$$

Since  $F_{n,d+1}$  computes **OV**, it follows that for every input vector  $z \in \{0, 1\}^d$  of Hamming weight  $d/2$ ,  $F'_n$  on input  $z$  outputs 1 if and only if there is some  $i$  such that  $\langle \overline{1x'_i}, 0z \rangle = 0$ . Note that since both  $\overline{1x'_i}$  and  $z$  have Hamming weight exactly  $d/2$ , we have  $\langle \overline{1x'_i}, 0z \rangle = 0$  if and only if  $z = x_i$ .

Let  $f : \{0, 1\}^d \rightarrow \{0, 1\}$  be any function that is consistent with the list of hard pairs  $\{(x_i, y_i)\}$  (from Corollary 8 in case 1, and our claim in case 2). By our choice of  $x_i$ 's, it follows that for all  $z \in \{0, 1\}^d$  of Hamming weight  $d/2$ ,  $F'_n(z) = 1$  if and only if  $f(z) = 1$ . By our choice of  $f$ , we must have

$$\text{size}(F'_n) \geq \min\{\Omega(2^d/(d^{1/2} \log d)), \Omega(n/\log d)\}, \quad (3)$$

depending on whether  $\binom{d}{d/2} \leq n-1$  or not (case 1 or case 2). Combining (2) and (3), we infer that

$$\ell \geq \Omega(n \cdot \min\{2^d/(d^{1/2} \log d), n/\log d\}). \quad (4)$$

Therefore the overall lower bound on formula size is  $s \geq \Omega\left(\min\left\{\frac{n^2}{\log d}, \frac{n \cdot 2^d}{d^{1/2} \log d}\right\}\right)$ .  $\blacktriangleleft$

### Remark on a Red-Blue Variant of **OV**

In the literature, **OV** is sometimes posed in a different form, where half of the vectors are colored red, half are colored blue, and we wish to find a red-blue pair which is orthogonal. Calling this form **OV'**, we note that **OV'** also exhibits the same lower bound up to constant factors. Given an algorithm/formula/circuit  $A$  for computing **OV'** on  $2n$  vectors ( $n$  of which are red, and  $n$  of which are blue), it is easy to verify that an algorithm/formula/circuit for **OV** on  $n$  vectors results by simply putting two copies of the set of vectors in the red and blue parts. Thus our lower bounds hold for the red-blue variant as well.



## 2.2 Lower Bound for Branching Programs

Recall that a branching program of size  $S$  on  $n$  variables is a directed acyclic graph  $G$  on  $S$  nodes, with a distinguished start node  $s$  and exactly two sink nodes, labeled 0 and 1 respectively. All non-sink nodes are labeled with a variable  $x_i$  from  $\{x_1, \dots, x_n\}$ , and have one outgoing edge labeled  $x_i = 1$  and another outgoing edge labeled  $x_i = 0$ . The branching program  $G$  evaluated at an input  $(a_1, \dots, a_n) \in \{0, 1\}^n$  is the subgraph obtained by only including edges of the form  $x_i = a_i$ , for all  $i = 1, \dots, n$ . Note that after such an evaluation, the remaining subgraph has a unique path from the start node  $s$  to a sink; the sink reached on this unique path (be it 0 or 1) is defined to be the output of  $G$  on  $(a_1, \dots, a_n)$ .

► **Reminder of Theorem 3.** **OV** on  $n$  vectors in  $d$  dimensions does not have branching programs of size  $O(\min\{n^2, n \cdot 2^d / (d^{1/2})\} / (\log(nd) \log(d)))$ , for all sufficiently large  $n, d$ .

**Proof Sketch.** The proof is similar to Theorem 2; here we focus on the steps of the proof that are different. Let  $G$  be a branching program with  $S$  nodes computing **OV** on  $n$  vectors in  $d + 1$  dimensions. Each node of  $G$  reads a single input bit from one of the input vectors; thus there is an input vector  $v_{i^*}$  that is read only  $O(S/n)$  times in the entire branching program  $G$ .

We will assign all variables other than the  $d + 1$  variables that are part of  $v_{i^*}$ . Using the same encoding as Theorem 2, by assigning  $n - 1$  other vectors, we can implement a function  $f : \{0, 1\}^d \rightarrow \{0, 1\}$  that is hard for branching programs to compute on the  $d$ -bit inputs in  $\binom{[d]}{[d/2]}$ . In particular, we substitute  $(d + 1)$ -bit vectors which represent inputs from  $f^{-1}(1) \cap \binom{[d]}{[d/2]}$  for all  $n - 1$  input vectors different from  $v_{i^*}$ . For each of these assignments, we can reduce the size of the branching program accordingly: for each input bit  $x_j$  that is substituted with the bit  $a_j$ , we remove all edges with the label  $x_j = \neg a_j$ , so that every node labeled  $x_j$  now has outdegree 1. After the substitution, two properties hold:

1. There is a hard function  $f$  such that the minimum size  $T$  of a branching program computing  $f$  on the  $n - 1$  inputs satisfies  $T \log_2(T) \geq \Omega(\min\{\binom{[d]}{[d/2]}, n\} / \log(d))$ . To see that such an  $f$  exists, we again use a counting argument. First note there are  $d^T \cdot 2^{\Theta(T \log(T))}$  branching programs of size  $T$  on  $d$  inputs (there are  $d^T$  choices for the node labels, and  $2^{\Theta(T \log(T))}$  choices for the remaining graph on  $T$  nodes). In contrast, there are at least  $2^{\min\{\binom{[d]}{[d/2]}, n-1\}}$  choices for the hard function  $f$ 's values on  $d$ -bit inputs of Hamming weight  $d/2$ . Therefore there is a function  $f$  such that  $d^T \cdot 2^{\Theta(T \log(T))} \geq 2^{\min\{\binom{[d]}{[d/2]}, n-1\}}$ , or

$$T + \Theta(T \log(T)) \geq \min \left\{ \binom{[d]}{[d/2]}, n - 1 \right\} / \log_2(d).$$

2. The minimum size of a branching program computing a function  $f : \{0, 1\}^d \rightarrow \{0, 1\}$  on the remaining  $d$  bits of input is at most  $O(S/n)$ . This follows because every node  $v$  with outdegree 1 can be removed from the branching program without changing its functionality: for every arc  $(u, v)$  in the graph, we can replace it with the arc  $(u, v')$ , where  $(v, v')$  is the single edge out of  $v$ , removing the node  $v$ .

Combining these two points, we have  $(S/n) \cdot \log(S/n) \geq \Omega\left(\min\left\{\binom{[d]}{[d/2]}, n\right\} / \log(d)\right)$ , or

$$S \geq \Omega\left(\frac{\min\{n \binom{[d]}{[d/2]}, n^2\}}{\log(S/n) \cdot \log(d)}\right).$$

Since  $S \leq n^2 d$ , we have

$$S \geq \Omega\left(\frac{\min\{n \binom{[d]}{[d/2]}, n^2\}}{\log(nd) \cdot \log(d)}\right) \geq \Omega\left(\frac{\min\{n \cdot 2^d / d^{1/2}, n^2\}}{\log(nd) \cdot \log(d)}\right).$$

This concludes the proof. ◀

## 2.3 Formulas With Symmetric Gates

We will utilize a lower bound on the number of functions computable by symmetric-gate formulas with a small number of wires:

► **Lemma 9.** *There are  $n^{O(w)}$  symmetric-gate formulas with  $w$  wires and  $n$  inputs.*

**Proof.** There is an injective mapping from the set of trees of unbounded fan-in and  $w$  wires into the set of binary trees with at most  $2w$  nodes: simply replace each node of fan-in  $k$  with a binary tree of at most  $2k$  nodes. The number of such binary trees is  $O(4^{2w})$  (by upper bounds on Catalan numbers). This counts the number of “shapes” for the symmetric formula; we also need to count the possible gate assignments. There are  $2^{k+1}$  symmetric functions on  $k$  inputs. So for a symmetric-gate formula with  $g$  gates, where the  $i$ th gate has fan-in  $w_i$  for  $i = 1, \dots, g$ , the number of possible assignments of symmetric functions to its gates is  $\prod_{i=1}^g 2^{w_i+1} = 2^{g+\sum_i w_i} = 2^{g+w}$ . There are at most  $w$  leaves, and there are  $n^w$  ways to choose the variables read at each leaf. Since  $g \leq w$ , we conclude that there are at most  $4^{2w} \cdot 2^{2w} \cdot n^w \leq n^{O(w)}$  symmetric-gate formulas with  $w$  wires. ◀

► **Reminder of Theorem 5.** *Every formula computing  $\mathbf{OV}$  composed of arbitrary symmetric functions with unbounded fan-in needs at least  $\Omega(\min\{n^2/(\log d), n \cdot 2^d/(d^{1/2} \log d)\})$  wires, for all  $n$  and  $d$ .*

**Proof.** (Sketch) With Lemma 9 in hand, the proof is quite similar to the previous lower bounds, so we just sketch the ideas. Let  $F$  be a symmetric-gate formula for computing  $\mathbf{OV}$  with unbounded fan-in and  $w$  wires. Let  $w_i$  be the number of wires touching inputs and  $w_g$  be the number of wires that do not touch inputs. Since  $F$  is a formula, we have (by a simple induction argument) that  $w_i \geq w_g$ , thus

$$w \leq 2w_i. \tag{5}$$

As before, each leaf of the formula is labeled by an input from one of the input  $n$  vectors; in this way, every leaf is “owned” by one of the  $n$  input vectors. We will substitute a 0/1 variable assignment to all vectors, except the vector  $\vec{z}^*$  which owns the fewest leaves. This gives a 0/1 assignment to all but  $O(w_i/n)$  of the  $w_i$  wires that touch inputs.

After any such variable assignment, we can simplify  $F$  as follows. For every symmetric-function gate  $g$  which has  $w_g$  input wires with  $k$  wires assigned 0/1, we can replace  $g$  with a symmetric function  $g'$  that has only  $w_g - k$  inputs, and no input wires assigned 0/1 (a partial 0/1 assignment to a symmetric function just yields another symmetric function on a smaller set of inputs). If  $g'$  is equivalent to a constant function itself, then we remove it from the formula and substitute its output wire with that constant, repeating the process on the gates that use the output of  $g$  as input. When this process completes, our new formula  $F'$  has  $d$  inputs and no wires that are assigned constants. So  $F'$  has  $O(w_i/n)$  wires touching inputs, and therefore by (5) the total number of wires in  $F'$  is  $O(w/n)$ .

As described earlier, the  $n - 1$  vectors we assign can implement  $2^{\min\{n-1, \binom{d}{d/2}\}}$  different functions on  $d$ -bit inputs, but there are at most  $d^{O(w/n)}$  functions computable by the symmetric formula remaining, by Lemma 9. Thus the number of wires  $w$  must satisfy  $d^{O(w/n)} \geq 2^{\min\{n-1, \binom{d}{d/2}\}}$ , or

$$w \geq \Omega(\min\{n^2, n \cdot 2^d/(d^{1/2})\}/(\log d)).$$

This completes the proof. ◀

### 3 Small Formulas for OV in the Average Case

Recall that for  $p \in (0, 1)$  and for a fixed  $n$  and  $d$ , we say that  $\mathbf{OV}(p)_{n,d}$  is the distribution of  $\mathbf{OV}$  instances where all bits of the  $n$  vectors from  $\{0, 1\}^d$  are chosen independently, set to 1 with probability  $p$  and 0 otherwise. We will often say that a vector is “sampled from  $\mathbf{OV}(p)$ ” if each of its bits are chosen independently in this way. We would like to understand how efficiently  $\mathbf{OV}(p)_{n,d}$  can be solved on almost all instances (i.e., with probability  $1 - o(1)$ ), for every  $n$  and  $d$ .

► **Reminder of Theorem 6.** *For every  $p \in (0, 1)$ , and every  $n$  and  $d$ , there is an  $\text{AC}^0$  formula of size  $n^{2-\varepsilon_p}$  that correctly answers all but a  $o_n(1)$  fraction of  $\mathbf{OV}(p)_{n,d}$  instances on  $n$  vectors and  $d$  dimensions, for an  $\varepsilon_p > 0$  such that  $\varepsilon_p \rightarrow 1$  as  $p \rightarrow 1$ .*

**Proof.** Let  $\varepsilon > 0$  be sufficiently small in the following. First, we observe that  $\mathbf{OV}(p)_{n,d}$  is very easy, unless  $d$  is close to  $(2/\log_2(1/(1-p^2)))\log_2(n)$ . In particular, for dimensionality  $d$  that is significantly smaller (or larger, respectively) than this quantity, all but a  $o(1)$  fraction of the  $\mathbf{OV}(p)_{n,d}$  instances are “yes” (or “no”, respectively). To see this, note that two randomly chosen  $d$ -dimensional vectors under the  $\mathbf{OV}(p)_{n,d}$  distribution are orthogonal with probability  $(1-p^2)^d$ . For  $d = (2/\log_2(1/(1-p^2)))\log_2(n)$ , a random pair is orthogonal with probability

$$(1-p^2)^{(2/\log_2(1/(1-p^2)))\log_2(n)} = 1/n^2.$$

Thus an  $\mathbf{OV}(p)_{n,d}$  instance with  $n$  vectors has nontrivial probability of being a yes instance for  $d$  approximately  $(2/\log_2(1/(1-p^2)))\log_2(n)$ .

Therefore if  $d > (2/\log_2(1/(1-p^2)) + \varepsilon)\log_2(n)$ , or  $d < (2/\log_2(1/(1-p^2)) - \varepsilon)\log_2(n)$ , then the random instance is either almost surely a “yes” instance, or almost surely a “no” instance, respectively. These comparisons could be done with the quantities  $(2/\log_2(1/(1-p^2)) - \varepsilon)\log_2(n)$  and  $(2/\log_2(1/(1-p^2)) + \varepsilon)\log_2(n)$  (which can be hard-coded in the input) with a  $\text{poly}(d, \log n)$ -size branching program, which can output 0 and 1 respectively if this is the case.<sup>4</sup>

From here on, assume that

$$d \in [(2/\log_2(1/(1-p^2)) - \varepsilon)\log_2(n), (2/\log_2(1/(1-p^2)) + \varepsilon)\log_2(n)].$$

Note that for  $p$  sufficiently close to 1, the dimensionality  $d$  is  $\delta \log n$  for a small constant  $\delta > 0$  that is approaching 0. Thus in the case of large  $p$ , the  $\text{AC}^0$  formula given in Proposition 4 has sub-quadratic size. In particular, the size is

$$O(n \cdot 2^d \cdot d) \leq n^{1+2/\log_2(1/(1-p^2))+o(1)}. \quad (6)$$

For  $p \geq 0.867 > \sqrt{3/4}$ , this bound is sub-quadratic. For smaller  $p$ , we will need a more complex argument.

Suppose  $u, v \in \{0, 1\}^d$  are randomly chosen according to the distribution of  $\mathbf{OV}(p)$  (we will drop the  $n, d$  subscript, as we have fixed  $n$  and  $d$  at this point).

We now claim that, conditioned on the event that  $u, v$  is an orthogonal pair, both  $u$  and  $v$  are expected to have between  $(p/(1+p) - \varepsilon)d$  and  $(p/(1+p) + \varepsilon)d$  ones, with  $1 - o(1)$

<sup>4</sup> As usual,  $\text{poly}(m)$  refers to an unspecified polynomial of  $m$  of fixed degree.

probability. The event that both  $u[i] = v[i] = 1$  holds with probability  $p^2$ ; conditioned on this event never occurring, we have

$$\begin{aligned}\Pr[u[i] = 0, v[i] = 0 \mid \neg(u[i] = v[i] = 1)] &= (1-p)^2/(1-p^2), \\ \Pr[u[i] = 1, v[i] = 0 \mid \neg(u[i] = v[i] = 1)] &= p(1-p)/(1-p^2), \\ \Pr[u[i] = 0, v[i] = 1 \mid \neg(u[i] = v[i] = 1)] &= p(1-p)/(1-p^2).\end{aligned}$$

Hence the expected number of ones in  $u$  (and in  $v$ ) is only  $p(1-p)d/(1-p^2) = pd/(1+p)$ , and the number of ones is within  $(-\varepsilon d, \varepsilon d)$  of this quantity with probability  $1 - o(1)$ . (For example, in the case of  $p = 1/2$ , the expected number of ones is  $d/3$ , while a typical vector has  $d/2$  ones.)

Say that a vector  $u$  is *light* if it has at most  $(p/(1+p) + \varepsilon)d$  ones. It follows from the above discussion that, conditioned on an  $\mathbf{OV}(p)$  instance being a “yes” instance, there is an orthogonal pair with two light vectors, with probability  $1 - o(1)$ . Since the expected number of ones is  $pd$ , the probability that a randomly chosen  $u$  is light is

$$\begin{aligned}\Pr\left[u \text{ has } \leq \left(\frac{p}{1+p} + \varepsilon\right)d = pd\left(1 - \frac{p}{p+1} + \frac{\varepsilon}{p}\right) \text{ ones}\right] &\leq e^{-(p/(p+1) - \varepsilon/p)^2 pd/2} \\ &= e^{-p^3 d / (2(p+1)^2) + \Theta_p(\varepsilon)d},\end{aligned}$$

by a standard Chernoff tail bound (see Theorem 11 in Appendix A). So with high probability, there are at most  $n \cdot e^{-p^3 d / (2(p+1)^2) + \Theta_p(\varepsilon)d} = n^{1-\alpha}$  light vectors in an  $\mathbf{OV}(p)$  instance, where

$$\alpha = \frac{\log_2(e) \cdot p^3}{(p+1)^2 \log_2(1/(1-p^2))} + \Theta_p(\varepsilon) / \log_2(1/(1-p^2)).$$

Divide the  $n$  vectors of the input arbitrarily into  $n^{1-\alpha(1-\varepsilon)}$  groups  $G_1, \dots, G_{n^{1-\alpha(1-\varepsilon)}}$ , of  $O(n^{\alpha(1-\varepsilon)})$  vectors each. WLOG, suppose an orthogonal pair  $u, v$  lies in different groups  $u \in G_i$  and  $v \in G_j$ , with  $i \neq j$  (note that, conditioned on there being an orthogonal pair, this event also occurs with  $1 - o(1)$  probability). Since every vector is independently chosen, and given that  $\Pr_v[v \text{ is light}] \leq 1/n^\alpha$ , note that

$$\Pr_{v_1, \dots, v_{n^{\alpha(1-\varepsilon)}}} [\text{all } v_i \text{ in group } G_a \text{ are not light}] \geq (1 - 1/n^\alpha)^{n^{\alpha(1-\varepsilon)}} \geq 1 - 1/n^{\varepsilon\alpha},$$

for every group  $G_a$ . Thus the groups  $G_i$  and  $G_j$  have at most one light vector with probability  $1 - o(1)$ .

Let  $Light(v)$  be the function which outputs 1 if and only if the  $d$ -bit input vector  $v$  is light. Since every symmetric function has  $\text{poly}(d)$ -size formulas [29],  $Light(v)$  also has  $\text{poly}(d)$ -size formulas. We can now describe our formula for  $\mathbf{OV}(p)$ , in words:

Take the OR over all  $n^{2-2\alpha(1-\varepsilon)}$  pairs  $(i, j) \in [n^{1-\alpha(1-\varepsilon)}]^2$  with  $i < j$ :  
 Take the  $\neg$ OR over all  $k = 1, \dots, d$ , of the AND of two items:  
 1. The OR over all  $O(n^{\alpha(1-\varepsilon)})$  vectors  $u$  in group  $G_i$  of  $(Light(u) \wedge u[k])$ .  
 2. The OR over all  $O(n^{\alpha(1-\varepsilon)})$  vectors  $v$  in group  $G_j$  of  $(Light(v) \wedge v[k])$ .

To see that this works, we observe:

- If there is an orthogonal pair  $u, v$  in the instance, then recall that with probability  $1 - o(1)$ , (a)  $u$  and  $v$  are light, (b)  $u$  and  $v$  appear in different groups  $G_i$  and  $G_j$ , and (c) there are no other light vectors in  $G_i$  and no other light vectors in  $G_j$ . Thus the inner ORs

- over the group  $G_i$  (and respectively  $G_j$ ) will only output the bits of the vector  $u$  (and respectively  $v$ ). Thus the above formula, by guessing the pair  $(i, j)$ , and checking over all  $k = 1, \dots, d$  that  $(u[k] \wedge v[k])$  is *not* true, will find that  $u, v$  are orthogonal, and output 1.
- If there is no orthogonal pair, then we claim that the formula *always* outputs 0. Suppose the formula outputs 1. Then there is some  $(i, j) \in [n^{1-\alpha(1-\varepsilon)}]^2$  such that the inner product of two vectors  $V_i$  and  $W_j$  is 0, where  $V_i$  is the OR of *all* light vectors in group  $G_i$  and  $W_j$  is the OR of all light vectors in group  $G_j$ . But for these two vectors to have zero inner product, it must be that *all* pairs of light vectors (one from  $G_i$  and one from  $G_j$ ) are orthogonal to each other. Thus there is an orthogonal pair in the instance.

Using the  $\text{poly}(d)$ -size formulas for *Light*, the DeMorgan formula has size

$$O(n^{2-2\alpha(1-\varepsilon)} \cdot d \cdot n^{\alpha(1-\varepsilon)} \cdot \text{poly}(d)) \leq O(n^{2-\alpha(1-\varepsilon)} \cdot \text{poly}(d)). \tag{7}$$

Substituting in the value for  $\alpha$ , the exponent becomes

$$2 - \frac{p^3(1-\varepsilon)}{(p+1)^2 \ln(1/(1-p^2))} + \Theta_p(\varepsilon)/\ln(1/(1-p^2)).$$

Recalling that we are setting  $\varepsilon$  to be arbitrarily small (its value only affects the  $o(1)$  probability of error), the formula size is

$$n^{2 - \frac{p^3}{2(p+1)^2 \ln(1/(1-p^2))} + o(1)}.$$

Observe that our formula can in fact be made into an  $\text{AC}^0$  formula of similar size; this is easy to see except for the  $\text{poly}(d)$ -size formula for *Light*. But for  $d = O(\log n)$ , any formula of  $\text{poly}(\log n)$ -size on  $O(\log n)$  bits can be converted into an  $\text{AC}^0$  circuit of depth  $c/\varepsilon$  and size  $2^{(\log n)^\varepsilon}$ , for some constant  $c \geq 1$  and any desired  $\varepsilon > 0$ .

The final formula is the minimum of the formulas of (6) and (7). For every fixed  $p \in (0, 1]$ , we obtain a bound of  $n^{2-\varepsilon_p}$  for an  $\varepsilon_p > 0$ . ◀

## 4 Conclusion

Let us highlight two interesting directions for future work.

### Lower Bounds for AC0 Circuits?

While we give fairly strong lower bounds for Boolean formulas, our results seem to leave open the following compelling conjecture about the difficulty of **OV** with low-depth circuits:

► **Conjecture 10.** *There is a  $\delta > 0$  such that **OV** is not computable by depth-3  $\text{AC}^0$  circuits with  $n^{2-\varepsilon}$  wires.*

The obvious translation of a depth-3 circuit into a formula would blow up the size too much, so our formula lower bounds do not (easily) extend to resolve the above conjecture. Typical random restriction methods (e.g., [25]) appear to be too coarse to handle such conjectures, but perhaps deterministic restriction methods (e.g., [21]) can.

### Generalizations of OV?

It is important to note that the *largest* known lower bound for branching programs computing any explicit function is due to Neciporuk [32] from 1966, and is only  $\Omega(N^2/\log^2 N)$  for inputs of length  $N$ . A similar statement holds for Boolean formulas over the full binary basis (see for

example [28]). Our lower bounds for **OV** match these bounds up to polylogarithmic factors. Thus it would be a significant breakthrough to generalize our results to other problems believed to require *cubic* time, such as:

3-ORTHOGONAL VECTORS (3-**OV**)  
 Given:  $n$  vectors  $v_1, \dots, v_n \in \{0, 1\}^d$   
 Decide: Are there  $i, j, k$  such that  $\sum_{\ell=1}^d v_i[\ell] \cdot v_j[\ell] \cdot v_k[\ell] = 0$ ?

It is known that the Strong Exponential Time Hypothesis also implies that 3-**OV** requires  $n^{3-o(1)}$  for dimensionality  $d = \omega(\log n)$  [37, 4].

---

### References

- 1 Pairwise comparison of bit vectors, January 20, 2017. URL: <https://cstheory.stackexchange.com/questions/37361/pairwise-comparison-of-bit-vectors>.
- 2 Amir Abboud, Arturs Backurs, Thomas Dueholm Hansen, Virginia Vassilevska Williams, and Or Zamir. Subtree Isomorphism Revisited. In *SODA*, pages 1256–1271, 2016.
- 3 Amir Abboud, Arturs Backurs, and Virginia Vassilevska Williams. Tight Hardness Results for LCS and Other Sequence Similarity Measures. In *FOCS*, pages 59–78, 2015.
- 4 Amir Abboud and Virginia Vassilevska Williams. Popular Conjectures Imply Strong Lower Bounds for Dynamic Problems. In *FOCS*, pages 434–443, 2014.
- 5 Amir Abboud, Virginia Vassilevska Williams, and Joshua Wang. Approximation and fixed parameter subquadratic algorithms for radius and diameter in sparse graphs. In *SODA*, pages 377–391. Society for Industrial and Applied Mathematics, 2016.
- 6 Amir Abboud, Virginia Vassilevska Williams, and Oren Weimann. Consequences of Faster Alignment of Sequences. In *ICALP*, pages 39–51, 2014.
- 7 Amir Abboud, Richard Ryan Williams, and Huacheng Yu. More Applications of the Polynomial Method to Algorithm Design. In *SODA*, pages 218–230, 2015.
- 8 Thomas Dybdahl Ahle, Rasmus Pagh, Ilya P. Razenshteyn, and Francesco Silvestri. On the Complexity of Inner Product Similarity Join. In *PODS*, pages 151–164, 2016.
- 9 Arturs Backurs and Piotr Indyk. Edit Distance Cannot Be Computed in Strongly Subquadratic Time (unless SETH is false). In *STOC*, pages 51–58, 2015.
- 10 Arturs Backurs and Piotr Indyk. Which Regular Expression Patterns Are Hard to Match? In *FOCS*, pages 457–466, 2016.
- 11 Marshall Ball, Alon Rosen, Manuel Sabin, and Prashant Nalini Vasudevan. Average-Case Fine-Grained Hardness. *IACR Cryptology ePrint Archive*, 2017:202, 2017. URL: <http://eprint.iacr.org/2017/202>.
- 12 Allan Borodin. On Relating Time and Space to Size and Depth. *SIAM J. Comput.*, 6(4):733–744, 1977. doi:10.1137/0206054.
- 13 Karl Bringmann. Why Walking the Dog Takes Time: Fréchet Distance Has No Strongly Subquadratic Algorithms Unless SETH Fails. In *FOCS*, pages 661–670, 2014.
- 14 Karl Bringmann and Marvin Künnemann. Quadratic Conditional Lower Bounds for String Problems and Dynamic Time Warping. In *FOCS*, pages 79–97, 2015.
- 15 Karl Bringmann and Wolfgang Mulzer. Approximability of the discrete Fréchet distance. *JoCG*, 7(2):46–76, 2016.
- 16 Kevin Buchin, Maike Buchin, Maximilian Konzack, Wolfgang Mulzer, and André Schulz. Fine-grained analysis of problems on curves. In *EuroCG, Lugano, Switzerland*, 2016.
- 17 Massimo Cairo, Roberto Grossi, and Romeo Rizzi. New Bounds for Approximating Extremal Distances in Undirected Graphs. In *SODA*, pages 363–376, 2016.



- 18 Chris Calabro, Russell Impagliazzo, and Ramamohan Paturi. The Complexity of Satisfiability of Small Depth Circuits. In *Parameterized and Exact Complexity (IWPEC)*, pages 75–85, 2009.
- 19 Timothy M. Chan and Ryan Williams. Deterministic APSP, Orthogonal Vectors, and More: Quickly Derandomizing Razborov-Smolensky. In *SODA*, pages 1246–1255, 2016. doi:10.1137/1.9781611974331.ch87.
- 20 Krishnendu Chatterjee, Wolfgang Dvořák, Monika Henzinger, and Veronika Loitzenbauer. Model and Objective Separation with Conditional Lower Bounds: Disjunction is Harder than Conjunction. In *LICS*, pages 197–206, 2016.
- 21 Shiva Chaudhuri and Jaikumar Radhakrishnan. Deterministic restrictions in circuit complexity. In *STOC*, pages 30–36. ACM, 1996.
- 22 Lijie Chen and Ryan Williams. An Equivalence Class for Orthogonal Vectors. In *SODA*, page to appear, 2019.
- 23 Jacob Evald and Søren Dahlgaard. Tight Hardness Results for Distance and Centrality Problems in Constant Degree Graphs. *CoRR*, abs/1609.08403, 2016. arXiv:1609.08403.
- 24 Jiawei Gao, Russell Impagliazzo, Antonina Kolokolova, and R. Ryan Williams. Completeness for First-Order Properties on Sparse Structures with Algorithmic Applications. In *SODA*, pages 2162–2181, 2017.
- 25 Johan Håstad. Almost Optimal Lower Bounds for Small Depth Circuits. In *Proceedings of the 18th Annual ACM Symposium on Theory of Computing*, pages 6–20, 1986.
- 26 Costas S. Iliopoulos and Jakub Radoszewski. Truly Subquadratic-Time Extension Queries and Periodicity Detection in Strings with Uncertainties. In *27th Annual Symposium on Combinatorial Pattern Matching, CPM 2016, June 27-29, 2016, Tel Aviv, Israel*, pages 8:1–8:12, 2016.
- 27 Russell Impagliazzo and Ramamohan Paturi. On the Complexity of k-SAT. *J. Comput. Syst. Sci.*, 62(2):367–375, 2001.
- 28 Stasys Jukna. *Boolean Function Complexity: Advances and Frontiers*. Springer-Verlag, 2012.
- 29 V. M. Khrapchenko. The complexity of the realization of symmetrical functions by formulae. *Mathematical notes of the Academy of Sciences of the USSR*, 11(1):70–76, 1972.
- 30 Marvin Künnemann, Ramamohan Paturi, and Stefan Schneider. On the Fine-grained Complexity of One-Dimensional Dynamic Programming. *CoRR*, abs/1703.00941, 2017. arXiv:1703.00941.
- 31 Nancy A. Lynch. Log Space Recognition and Translation of Parenthesis Languages. *J. ACM*, 24(4):583–590, 1977. doi:10.1145/322033.322037.
- 32 E. I. Nechiporuk. On a Boolean function. *Doklady of the Academy of Sciences of the USSR*, 169(4):765–766, 1966. English translation in *Soviet Mathematics Doklady* 7:4, pages 999–1000.
- 33 Paul Pritchard. A Fast Bit-Parallel Algorithm for Computing the Subset Partial Order. *Algorithmica*, 24(1):76–86, 1999.
- 34 Liam Roditty and Virginia Vassilevska Williams. Fast approximation algorithms for the diameter and radius of sparse graphs. In *STOC*, pages 515–524, 2013.
- 35 Michael Wehar. Intersection Non-Emptiness for Tree-Shaped Finite Automata. Available at <http://michaelwehar.com/documents/TreeShaped.pdf>, February 2016.
- 36 Richard Ryan Williams. Strong ETH Breaks With Merlin and Arthur: Short Non-Interactive Proofs of Batch Evaluation. In *31st Conference on Computational Complexity, CCC 2016, May 29 to June 1, 2016, Tokyo, Japan*, pages 2:1–2:17, 2016. doi:10.4230/LIPIcs.CCC.2016.2.
- 37 Ryan Williams. A new algorithm for optimal 2-constraint satisfaction and its implications. *Theor. Comput. Sci.*, 348(2-3):357–365, 2005. See also ICALP’04.
- 38 Ryan Williams and Huacheng Yu. Finding orthogonal vectors in discrete structures. In *SODA*, pages 1867–1877, 2014. doi:10.1137/1.9781611973402.135.



## A Chernoff Bound

We use the following standard tail bound:

► **Theorem 11.** *Let  $p \in (0, 1)$  and let  $X_1, \dots, X_d \in \{0, 1\}$  be independent random variables, such that for all  $i$  we have  $\Pr[X_i = 1] = p$ . Then for all  $\delta \in (0, 1)$ ,*

$$\Pr \left[ \sum_i X_i < (1 - \delta)pd \right] \leq e^{-\delta^2 pd/2}.$$

## B DeMorgan Formulas into Branching Programs

Here we describe at a high level how to convert a DeMorgan formula (over AND, OR, NOT) of size  $s$  into a branching program of size  $O(s)$ . Our construction is similar to the log-space algorithm for formula evaluation of Lynch [31], except we have to be extremely careful with the time and space efficiency to get a branching program with only  $O(s)$  total nodes (corresponding to  $\log_2(s) + O(1)$  space).

Our branching program will perform an in-order traversal of the DeMorgan formula, maintaining a counter (from 1 to  $s$ ) of the current node being visited in the formula. The branching program begins at the root (output) of the formula. If the current node is a leaf, its value  $b$  is returned to the parent node. If the current node is not a leaf, the branching program recursively evaluates its left child (storing no memory about the current node).

The left child returns a value  $b$ . If the current node is an AND and  $b = 0$ , or the current node is an OR and  $b = 1$ , the branching program propagates the bit  $b$  up the tree (moving up to the parent). If the current node is a NOT, then the branching program moves to the parent with the value  $\neg b$ .

If none of the above cases hold, then the branching program erases the value  $b$ , and recursively evaluates the right child, which returns a value  $b$ . This value is simply propagated up the tree (note the fact that we visited the right child means that we know what the left child's value was).

Observe that we only hold the current node of the formula in memory, as well as  $O(1)$  extra bits.



# SOS Lower Bounds with Hard Constraints: Think Global, Act Local

**Pravesh K. Kothari**

Department of Computer Science, Princeton University and Institute for Advanced Study,  
Princeton, USA  
kothari@cs.princeton.edu

**Ryan O’Donnell<sup>1</sup>**

Department of Computer Science, Carnegie Mellon University, Pittsburgh, USA  
odonnell@cs.cmu.edu

**Tselil Schramm<sup>2</sup>**

Department of Computer Science, Harvard and MIT, Cambridge, USA  
tselil@mit.edu

---

## Abstract

Many previous Sum-of-Squares (SOS) lower bounds for CSPs had two deficiencies related to global constraints. First, they were not able to support a “cardinality constraint”, as in, say, the Min-Bisection problem. Second, while the pseudoexpectation of the objective function was shown to have some value  $\beta$ , it did not necessarily actually “satisfy” the constraint “objective =  $\beta$ ”. In this paper we show how to remedy both deficiencies in the case of random CSPs, by translating *global* constraints into *local* constraints. Using these ideas, we also show that degree- $\Omega(\sqrt{n})$  SOS does not provide a  $(\frac{4}{3} - \varepsilon)$ -approximation for Min-Bisection, and degree- $\Omega(n)$  SOS does not provide a  $(\frac{11}{12} + \varepsilon)$ -approximation for Max-Bisection or a  $(\frac{5}{4} - \varepsilon)$ -approximation for Min-Bisection. No prior SOS lower bounds for these problems were known.

**2012 ACM Subject Classification** Theory of computation  $\rightarrow$  Semidefinite programming, Theory of computation  $\rightarrow$  Randomness, geometry and discrete structures

**Keywords and phrases** sum-of-squares hierarchy, random constraint satisfaction problems

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2019.49

**Related Version** <https://arxiv.org/abs/1809.01207>

**Acknowledgements** The authors very much thank Sangxia Huang and David Witmer for their contributions to the early stages of this research. Thanks also to Svante Janson for discussions concerning contiguity of random graph models. We also gratefully acknowledge comments on the manuscript from Johan Håstad as well as several anonymous reviewers.

---

<sup>1</sup> Some work performed at the Boğaziçi University Computer Engineering Department, supported by Marie Curie International Incoming Fellowship project number 626373. Also supported by NSF grants CCF-1618679, CCF-1717606. This material is based upon work supported by the National Science Foundation under grant numbers listed above. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author and do not necessarily reflect the views of the National Science Foundation (NSF).

<sup>2</sup> This work was partly supported by an NSF Graduate Research Fellowship (1106400), and also by a Simons Institute Fellowship.



© Pravesh Kothari, Ryan O’Donnell, and Tselil Schramm;  
licensed under Creative Commons License CC-BY

10th Innovations in Theoretical Computer Science (ITCS 2019).

Editor: Avrim Blum; Article No. 49; pp. 49:1–49:21

Leibniz International Proceedings in Informatics



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

## 1 Introduction

Consider the task of refuting a random 3SAT instance with  $n$  variables and  $50n$  clauses; i.e., certifying that it’s unsatisfiable (which it is, with very high probability). There is no known  $2^{o(n)}$ -time algorithm for this problem. An oft-cited piece of evidence for the exponential difficulty is the fact [9, 19] that the very powerful Sum-of-Squares (SOS) SDP hierarchy fails to refute such random 3SAT instances in  $2^{o(n)}$  time. Colloquially, degree- $\Omega(n)$  SOS “thinks” that the random 3SAT instance is satisfiable (with high probability).

But consider the following method of refuting satisfiability of a random  $50n$ -clause CNF  $\phi$ :

For all  $k \in \{0, 1, 2, \dots, n\}$ ,  
 refute “ $\phi$  is satisfiable by an assignment of Hamming weight  $k$ ”.

Could it be that  $O(1)$ -degree SOS succeeds in refuting random 3SAT instances in this manner? It seems highly unlikely, but prior to this work the possibility could not be ruled out.

### SOS lower bounds with Hamming weight constraints

Recall that the known SOS lower bounds for random 3SAT are actually stronger: they show degree- $\Omega(n)$  SOS thinks that random 3SAT instances are satisfiable even as *3XOR* (i.e., with every clause having an odd number of true literals). Hamming weight calculations are quite natural in the context of random 3XOR; indeed Grigoriev, Hirsch, and Pasechnik [10] showed that the *dynamic* degree-5 SOS proof system *can* refute random 3XOR instances by using integer counting techniques. Thus the above “refute solutions at each Hamming weight” strategy seems quite natural in the context of random CSPs.

In 2012, Yuan Zhou raised the question of proving strong SOS lower bounds for random 3XOR instances together with a global cardinality constraint such as  $\sum_i x_i = \frac{n}{2}$ . This would rule out the above refutation strategy. It is also a natural SOS challenge, seemingly combining the two strong SOS results known prior to 2012 – the bound for random 3XOR due to Grigoriev and Schoenebeck [9, 19] and the bound for Knapsack due to Grigoriev [8].

One may ask why the Grigoriev–Schoenebeck SOS lower bound doesn’t already satisfy  $\sum_i x_i = \frac{n}{2}$ . The difficulty is connected to the meaning of the word “satisfy”. One should think of the SOS Method as trying to find not just a satisfying assignment to a CSP, but more generally a *distribution* on satisfying assignments. The SOS algorithm finds a “degree- $d$  pseudodistribution” on satisfying assignments in  $n^{O(d)}$  time, provided one exists; roughly speaking, this means an object that “looks like” a distribution on satisfying assignment to all tests that are squared polynomials of degree at most  $d$ . For a random 3XOR instance with  $n$  variables and  $O(n)$  constraints, the Grigoriev–Schoenebeck degree- $\Omega(n)$  pseudodistribution indeed claims to have 100% of its probability mass on satisfying assignments. Furthermore, its assignments claim to give probability 50% to each of  $x_i = 0$  and  $x_i = 1$  for all  $i$ ; in other words, the “pseudoexpectation” of  $x_i$  is  $\frac{1}{2}$ , so the pseudoexpectation of  $\sum_i x_i$  is  $\frac{n}{2}$ . However, this *doesn’t* mean that the pseudodistribution “satisfies” the hard constraint  $\sum_i x_i = \frac{n}{2}$ . To actually “satisfy” this constraint, the expression  $\sum_i x_i$  must have *pseudovariance zero*; i.e., SOS must not only “think” it knows a distribution on 3XOR-satisfying assignments which has  $\sum_i x_i = \frac{n}{2}$  on average, it must think that *all* of these satisfying assignments have  $\sum_i x_i$  exactly  $\frac{n}{2}$ .

In this work we show how to upgrade any SOS lower bound for random CSPs based on  $t$ -wise uniformity so as to include the hard cardinality constraint  $\sum_i x_i = \frac{n}{2}$  (or indeed

$\sum_i x_i = \frac{n}{2} + k$  for any  $|k| = O(\sqrt{n})$ .<sup>3</sup> The idea is conceptually simple: just add a matching of 2XOR constraints,  $x_{2i-1} \neq x_{2i}$  for all  $1 \leq i \leq \frac{n}{2}$ .

### SOS lower bounds with exact objective constraints

A random 3AND CSP with  $n$  variables and  $m = \alpha n$  constraints (each an AND of 3 random literals) will have objective value  $\frac{1}{8} + \varepsilon$  with high probability, for  $\varepsilon$  arbitrarily small as a function of  $\alpha$ ; i.e., the best assignment will satisfy at most  $(\frac{1}{8} + \varepsilon)m$  constraints. On the other hand, it’s not too hard to show that the Grigoriev–Schoenebeck degree- $\Omega(n)$  pseudodistribution will give the objective function a pseudoexpectation of  $\frac{1}{4} \pm o(1)$ . (Roughly speaking, for almost all 3AND constraints, the SOS pseudodistribution will think it can obtain probability  $\frac{1}{4}$  on each of the 3XOR-satisfying assignments, and one of these, namely  $(1, 1, 1)$ , satisfies 3AND.) Thus it would appear that degree- $\Omega(n)$  SOS has an integrality gap of factor  $2 - \varepsilon$  on random 3AND instances.

But is this misleading? Suppose we solved the SOS SDP and it reported a solution with pseudoexpectation  $\frac{1}{4}$ . We might then “double-check” by re-running the SDP, together with an additional “equality constraint” specifying that the number of satisfied 3AND constraints is indeed  $\frac{1}{4}m$ .<sup>4</sup> As far as we know now, this run could return “infeasible”, actually *refuting* the possibility of  $\frac{1}{4}m$  constraints being satisfiable! Again, the issue is that under the Grigoriev–Schoenebeck SOS pseudodistribution, the objective function will have a pseudoexpectation like  $\frac{1}{4}$ , but will also have *nonzero pseudovariance*.

We show how to fix this issue – i.e., have the objective constraint be *exactly* SOS-satisfied – in the context of any SOS lower bound for random CSPs based on  $t$ -wise uniformity. Here we briefly express the idea of our solution, in the specific case of 3AND: We show that one can design a probability distribution  $\theta$  on  $r \times 3$  Boolean matrices such that two properties hold: (i)  $\theta$  is 2-wise uniform; (ii) for *every* outcome in the support of  $\theta$ , *exactly* a  $\frac{1}{4} - \varepsilon_r$  fraction of the  $r$  rows satisfy 3AND, where  $\varepsilon_r$  is an explicit positive constant depending on  $r$  that tends to 0 as  $r$  grows. We then use recent work [14] on constructing SOS lower bounds from  $t$ -wise uniform distributions to show that degree- $\Omega(n)$  SOS thinks it can “weakly satisfy” a random “distributional CSP” in which each constraint specifies that a random  $4r$ -tuple of variables should be distributed according to  $\theta$ . By “weak satisfaction”, we mean that SOS will at least think it can get a local distribution on each  $4r$ -tuple whose support is contained within  $\theta$ ’s support (and therefore always having exactly a  $\frac{1}{4} - \varepsilon_r$  fraction of rows satisfying 3AND). Now viewing each such tuple as the conjunction of  $r$  (random) 3AND constraints, we get that the SOS solution thinks it satisfies exactly a  $\frac{1}{4} - \varepsilon_r$  fraction of these constraints.

### Further consequences

Via our first result – satisfying global cardinality constraints – we open up the possibility of establishing SOS lower bounds for natural problems like Min- and Max-Bisection (by performing reductions within SOS, as in [21]). Previously, no such SOS integrality gaps were known (Guruswami, Sinop, and Zhou [11] had given an SOS integrality gap approaching  $\frac{11}{10}$  for the Balanced-Separator problem, which is like Min-Bisection but without a hard bisection constraint.) Under assumptions like  $\text{NP} \not\subseteq \bigcap_{\varepsilon > 0} \text{TIME}(2^{n^\varepsilon})$ , some hardness results were

<sup>3</sup> We also show in the full version that this is not too far from tight, in the sense that it is easier to refute XOR with Hamming weight constraints that are too imbalanced (if  $k = \omega(n^{1/4})$ ).

<sup>4</sup> Actually, it was recently observed that it is not clear we can definitely solve the associated SDP exactly [16, 18]. This does not affect the status of our lower bounds.

previously known: no PTAS for Min-Bisection (due to Khot [13]) and factor  $\frac{15}{16} + \varepsilon$  hardness for Max-Bisection (due to Holmerin and Khot [13], improving on the factor  $\frac{16}{17} + \varepsilon$  NP-hardness known for Max-Cut). However, in the context of SOS lower bounds, it makes sense to shoot for more: namely, hardness factors that are known subject to Feige’s R3SAT Hypothesis [7] (and similar hypotheses for random CSPs).

Feige himself [7] showed factor  $\frac{4}{3} - \varepsilon$  hardness for Min-Bisection under his hypothesis (with a quadratic size blowup). Also, it’s possible to show factor  $\frac{11}{12} + \varepsilon$  hardness for Max-Bisection (with linear size blowup) under Feige’s Hypothesis for 4XOR; this is arguably “folklore”, via the gadget techniques of Trevisan et al. [20] (see also [12, 17]). We are able to convert both of these results to SOS lower bounds, showing that degree- $\Omega(\sqrt{n})$  SOS fails to  $(\frac{4}{3} - \varepsilon)$ -approximate Min-Bisection, and degree- $\Omega(n)$  SOS fails to  $(\frac{5}{4} + \varepsilon)$ -approximate Min-Bisection. Our proof of the latter can also be modified to show that degree- $\Omega(n)$  SOS fails to  $(\frac{11}{12} + \varepsilon)$ -approximate Max-Bisection.

It is worth pointing out that the benefit of our second main result, the ability to enforce objective equality constraints exactly, also arises in these SOS Bisection lower bounds. For example, the  $(\frac{4}{3} - \varepsilon)$ -hardness for Min-Bisection is a kind of gadget reduction from random 3AND CSPs; showing that the “good cut” in the completeness case is an exact bisection relies on the “good assignment” in the 3AND instance satisfying exactly a  $\frac{1}{4}$  fraction of constraints.

## 1.1 Statement of main theorems

Recent work [3, 14] has established a general framework for showing lower bounds for SOS on random CSPs, using the idea of  $t$ -wise uniformity. The following is a fairly general example of what’s known:

► **Theorem 0** ([14]). *Let  $P : \{0, 1\}^k \rightarrow \{0, 1\}$  be a predicate, and suppose there is a  $(t - 1)$ -wise uniform distribution  $\nu$  on  $\{0, 1\}^k$  with  $\mathbb{E}_\nu[P] = \beta$ . Consider a random  $n$ -variable,  $m = \Delta n$ -constraint instance of  $\text{CSP}(P^\pm)$ , meaning that each constraint is  $P$  applied to  $k$  randomly chosen literals. Then with high probability, there is a degree- $\Omega\left(\frac{n}{\Delta^{2/(t-2)} \log \Delta}\right)$  SOS pseudodistribution  $\tilde{\mathbb{E}}[\cdot]$  with the following property:*

**Case 1:**  $\beta = 1$ . *In this case,  $\tilde{\mathbb{E}}[\cdot]$  satisfies all the CSP constraints as identities.*

**Case 2:**  $\beta < 1$ . *In this case,  $\mathbb{E}[\text{OBJ}(x)] = \beta \pm o(1)$ , where  $\text{OBJ}(x)$  denotes the objective value of the CSP.*

For example, the case of random 3SAT described in the previous section corresponds to  $P = \text{OR}_3$ ,  $t = 3$ ,  $\nu$  being the uniform distribution on triples satisfying  $\text{XOR}_3$ ,  $\beta = 1$ , and  $\Delta = 50$ ; the case of random 3AND has the same  $t$ ,  $\nu$ , and  $\Delta$ , but  $P = \text{AND}_3$  and  $\beta = \frac{1}{4}$ .

Our main theorems are now as follows:

► **Theorem 1.** *In the  $\beta = 1$  case of Theorem 0, one can additionally get the pseudodistribution  $\tilde{\mathbb{E}}$  to satisfy (with pseudovariance zero) the global bisection constraint  $\sum_{i=1}^n x_i = \frac{n}{2}$  (assuming  $n$  even). More generally, for any integer  $B \in [\frac{n}{2} - O(\sqrt{n}), \frac{n}{2} + O(\sqrt{n})]$ , we can ensure the pseudodistribution satisfies the global Hamming weight constraint  $\sum_{i=1}^n x_i = B$ .*

► **Theorem 2.** *In the  $\beta < 1$  case of Theorem 0, there exists a sequence of positive constants  $\varepsilon_r$  with  $\varepsilon_r \rightarrow 0$  such that for a random\*  $n$ -variable,  $m = \Delta n$ -constraint instance of  $\text{CSP}(P^\pm)$ , with high probability there is a degree- $\Omega_r\left(\frac{n}{\Delta^{2/(t-2)} \log \Delta}\right)$  SOS pseudodistribution  $\tilde{\mathbb{E}}$  which satisfies (with pseudovariance zero) the hard constraint “ $\text{OBJ}(x) = \beta - \varepsilon_r$ ”. Furthermore, we can also obtain cardinality constraints as in Theorem 1.*

In the full version, we show that Theorem 1 is not too far from tight, by demonstrating that random  $k$ -XOR instances become easier to refute when one imposes an imbalanced Hamming weight constraint  $\sum_{i=1}^n x_i = \frac{n}{2} \pm \omega(n^{1/4})$ .<sup>5</sup>

► **Remark.** In the above theorem we have written “random\*” with an asterisk because the random instance is not drawn precisely in the standard way. Rather, it is obtained by choosing  $m/r$  groups of random constraints, where in each group we fix a literal pattern and then choose  $r$  nonoverlapping constraints with this pattern. This technicality is an artifact of our proof; it seems likely that it is unnecessary. Indeed, it is possible that these two distributions on random hypergraphs are simply  $o(1)$ -close in total variation distance, at least when  $m = O(n)$ .<sup>6</sup> In any case, by alternate means (including the techniques from Theorem 1) we are able to show the following alternative result in Section 5.2: When  $m = o(n^{1.5})$ , with high probability a purely random instance of  $\text{CSP}(P^\pm)$  has an SOS pseudodistribution of the stated degree that exactly satisfies  $\text{OBJ}(x) = \beta - \varepsilon$  for some  $\varepsilon > 0$  that can be made arbitrarily small.

► **Remark.** Our proof of Theorem 2 only relies on the “Case 1,  $\beta = 1$ ” part of [14]’s Theorem 0. In fact, our Theorem 2 can actually be used to effectively *deduce* “Case 2,  $\beta < 1$ ” from “Case 1,  $\beta = 1$ ” in Theorem 0. This is of interest because [14]’s argument for Case 2 was not a black-box reduction from Case 1, but instead involved verifying a more technical expansion property in random graphs, as well as slightly reworking the proof of Case 2.

Finally, we obtain the following theorems concerning Bisection problems:

► **Theorem 3.** *For the Max-Bisection problem in a graph on  $n$  vertices, for  $d = \Omega(n)$ , the degree- $d$  Sum-of-Squares Method cannot obtain an approximation factor better than  $\frac{11}{12} - \varepsilon$  for any constant  $\varepsilon > 0$ .*

*For the Min-Bisection problem, for  $d' = \Omega(\sqrt{n})$ , the degree- $d'$  SOS Method cannot obtain an approximation factor better than  $\frac{4}{3} - \varepsilon$ , and for  $d = \Omega(n)$  the degree- $d$  SOS Method cannot obtain an approximation factor better than  $\frac{5}{4} - \varepsilon$ .*

The proofs are included in the full version.

## Organization of this paper

In Section 2, we provide some preliminaries and technical context for the study of CSPs and SOS. In Section 3, we extend the results of [14] to obtain lower bounds for CSPs with global cardinality constraints, proving Theorem 1. Section 4 shows how to construct local distributions over assignments to groups of disjoint predicates so that the number of satisfied constraints is always exactly the same, and Section 5 shows how to use such distributions to prove Theorem 2. In Section 5, one can also find a discussion of random vs. random\* CSPs. We wrap up with some concluding remarks and future directions in Section 6.

## 2 Preliminaries

### CSPs

A constraint satisfaction problem (CSP) is defined by an *alphabet*  $\Omega$  (usually  $\{0, 1\}$  or  $\{\pm 1\}$  in this paper) and a collection  $\mathcal{P}$  of *predicates*, each predicate being some  $P : \Omega^k \rightarrow \{0, 1\}$  (with different  $P$ ’s possibly having different *arities*,  $k$ ). An *instance*  $\mathcal{H}$  consists of a set  $V$  of

<sup>5</sup> We conjecture that Theorem 1 is tight, and that Hamming weight constraints with imbalance  $\omega(n^{1/2})$  already make  $k$ -XOR easier to refute.

<sup>6</sup> Thanks to Svante Janson for some observations in the direction of showing this.



$n$  variables, as well as  $m$  constraints. Each constraint  $h$  consists of a *scope*  $S$  and a predicate  $P \in \mathcal{P}$ , where  $S$  is a tuple of  $k$  distinct variables,  $k$  being the arity of  $P$ . An *assignment* gives a value  $x_i \in \Omega$  to the  $i$ th variable; it *satisfies* constraint  $h = (S, P)$  if  $P(x_{S_1}, \dots, x_{S_k}) = 1$ . We may sometimes write this as  $P(x_S) = 1$  for brevity. The associated *objective value* is the fraction of satisfied constraints,

$$\text{OBJ}(x) = \text{avg}_{h=(S,P) \in \mathcal{H}} \{P(x_S)\}.$$

Sometimes we are concerned with CSPs of the following type: the alphabet  $\Omega = \{\pm 1\}$  is Boolean, there is a single predicate  $P : \Omega^k \rightarrow \{0, 1\}$  (e.g.,  $P = \text{OR}_3$ , the 3-ary Boolean OR predicate), and the predicate set  $\mathcal{P}$  consists of all  $2^k$  versions of  $P$  in which inputs may be negated. We refer to this scenario as  $P$ -CSP with *literals*, denoted  $\text{CSP}(P^\pm)$ . For example, the case of  $P = \text{OR}_3$  is the classic “3SAT” CSP.

### Distributional CSPs

A *distributional CSP* is one where, rather than having a predicate associated with each scope, we have a probability distribution. More precisely, each *distributional constraint*  $h = (S, \nu)$  now consists of a scope  $S$  of some arity  $k$ , as well as a probability distribution  $\nu$  on  $\Omega^k$ . The optimization task involves finding a *global probability distribution*  $\mu$  on assignments. We say that  $\mu$  *satisfies* constraint  $h = (S, \nu)$  if the marginal  $\mu|_S$  of  $\mu$  on  $S$  is equal to  $\nu$ ; we say the distributional CSP is *satisfiable* if there is a  $\mu$  satisfying all constraints.

We may also say that  $\mu$  *weakly satisfies*  $h = (S, \nu)$  if  $\text{supp}(\mu|_S) \subseteq \text{supp}(\nu)$ . A “usual” (*predicative*, i.e., non-distributional) CSP can be viewed as a distributional CSP as follows: For each predicate  $P$ , select any distribution  $\nu_P$  whose support is exactly the satisfying assignments to  $P$ ; then the existence of a global assignment in the predicative CSP of objective value  $\beta$  is equivalent to the existence of a global probability distribution  $\mu$  that *weakly* satisfies a  $\beta$  fraction of constraints.

### Random CSPs

We are frequently concerned with CSPs chosen uniformly at random. Given a predicate set  $\mathcal{P}$ , a random CSP with  $n$  variables and  $m$  constraints is chosen as follows: For each constraint we first choose a random  $P \in \mathcal{P}$ . Supposing it has arity  $k$ , we then choose a uniformly random length- $k$  scope  $S$  from the  $n$  variables, and impose the constraint  $(S, P)$ . We can similarly define a random distributional CSP given a collection  $\mathcal{D}$  of distributions  $\nu$ . We remark that our choice of having exactly  $m$  constraints is not really essential, and not much would change if we had, e.g., a Poisson( $m$ ) number of random constraints, or if we chose each possible constraint independently with probability such that  $m$  constraints are expected.

### SOS

The SOS Method [5] can be thought of as an algorithmic technique for finding upper bounds on the best objective value achievable in a predicative or distributional CSP. For example, in a random 3SAT instance with  $m = 50n$ , it is very likely that every assignment  $x$  has  $\text{OBJ}(x) \leq \frac{7}{8} + o(1)$ ; ideally, the SOS Method could certify this, or could at least certify unsatisfiability, meaning an upper bound of  $\text{OBJ}(x) < 1$  for all assignments. The SOS Method has a tunable *degree* parameter  $d$ ; increasing  $d$  increases the effectiveness of the method, but also its run-time, which is essentially  $n^{O(d)}$  (though see [16, 18] for a more

precise discussion). In this work we are only concerned with showing negative results for the power of SOS. Showing that degree- $d$  SOS fails to certify a good upper bound on the maximum objective value is equivalent to showing that a *degree- $d$  pseudodistribution* exists under which the objective function has a large *pseudoexpectation*. We define these terms now.

For simplicity we restrict attention to CSPs with Boolean alphabet (either  $\Omega = \{0, 1\}$  or  $\Omega = \{\pm 1\}$ ), although it is straightforward to extend the definitions for larger alphabets.<sup>7</sup> The SOS method introduces *indeterminates*  $X_1, \dots, X_n$  associated to the CSP variables; intuitively, one thinks of them as standing for the outcome of a global assignment chosen from a supposed probability distribution on assignments. An associated *degree- $d$  pseudoexpectation* is a real-valued linear map  $\tilde{\mathbb{E}}$  on  $\mathbb{R}_{\leq d}[X_1, \dots, X_n]$  (the space of formal polynomials in  $X_1, \dots, X_n$  of degree at most  $d$ ) satisfying three properties:

1.  $\tilde{\mathbb{E}}[\text{multilin}(Q(X))] = \tilde{\mathbb{E}}[Q(X)]$ ; here  $\text{multilin}(Q(X))$  refers to the *multilinearization* of  $Q(X)$ , meaning the reduction mod  $X_i^2 = X_i$  (in case  $\Omega = \{0, 1\}$ ) or mod  $X_i^2 = 1$  (in case  $\Omega = \{\pm 1\}$ ).
2.  $\tilde{\mathbb{E}}[1] = 1$ ;
3.  $\tilde{\mathbb{E}}[Q(X)^2] \geq 0$  whenever  $\deg(Q) \leq d/2$ .

We tend to think of the first condition (as well as the linearity of  $\tilde{\mathbb{E}}$ ) as being “syntactically” enforced; i.e., given  $\tilde{\mathbb{E}}$ 's values on the multilinear monomials, its value on all polynomials is determined through multilinearization and linearity. It is not hard to show that every pseudoexpectation  $\tilde{\mathbb{E}}$  arises from a *signed probability distribution*  $\mu$ ; i.e., a (possibly negative) function  $\mu : \Omega^n \rightarrow \mathbb{R}$  with  $\sum_x \mu(x) = 1$ . We call this the associated *pseudodistribution*. Intuitively, we think of a degree- $d$  pseudodistribution as a “supposed” distribution on global assignments, which at least passes the tests in Item 3 above.

Given a CSP instance  $\mathcal{H}$ , if there is a degree- $d$  pseudodistribution with  $\tilde{\mathbb{E}}[\text{OBJ}(X)] \geq \beta$ , this means that the degree- $d$  SOS Method *fails* to certify an upper bound of  $\text{OBJ}(x) < \beta$  for the CSP. Informally, we say that degree- $d$  SOS “thinks” that there is a distribution on assignments under which the average objective value is at least  $\beta$ . Similarly, given a distributional CSP  $\mathcal{H}$ , if there is a degree- $d$  pseudodistribution in which  $\tilde{\mathbb{P}}[X_S = (a_1, \dots, a_k)] = \nu(a_1, \dots, a_k)$  for all constraints  $h = (S, \nu)$ , we say that degree- $d$  SOS “thinks” that  $\mathcal{H}$  is fully satisfiable. Here  $\tilde{\mathbb{P}}[X_S = (a_1, \dots, a_k)]$  means  $\tilde{\mathbb{E}}[1_{X_S=(a_1, \dots, a_k)}]$ , where  $1_{X_S=(a_1, \dots, a_k)}$  denotes the natural arithmetization of the 0-1 indicator as a degree- $k$  multilinear polynomial.

### Satisfaction of identities in SOS

Formally speaking, one says that a degree- $d$  pseudodistribution *satisfies an identity*  $Q(X) = b$  if  $\tilde{\mathbb{E}}[(Q(X) - b)R(X)] = 0$  for all polynomials  $R(X)$  of degree at most  $d - \deg(Q)$ . Note that this is *stronger* than simply requiring  $\tilde{\mathbb{E}}[Q(X)] = b$  (the  $R \equiv 1$  case). A great deal of this paper is concerned with precisely this distinction; it may be relatively easy to come up with a degree- $d$  pseudodistribution over  $\{0, 1\}^n$  satisfying, say,  $\tilde{\mathbb{E}}[\sum_i X_i] = \frac{n}{2}$ , but much harder to find one that “satisfies the identity  $\sum_i X_i = \frac{n}{2}$ ”. The terminology here is a little unfortunate; we will try to ameliorate things by introducing the following stronger phrase:

► **Definition 4.** We say that a degree- $d$  pseudodistribution satisfies identity  $Q(X) = b$  with *pseudovariance zero* if we have both  $\tilde{\mathbb{E}}[Q(X)] = b$  and *also*

$$\widetilde{\text{VAR}}[Q(X) - b] = \tilde{\mathbb{E}}[Q(X)^2] - b^2 = 0.$$

<sup>7</sup> Specifically, for each variable  $x$  and each alphabet element  $a \in \Omega$ , one introduces an indeterminate called  $1_{x=a}$  that is constrained as a  $\{0, 1\}$  value and is interpreted as the indicator of whether  $x$  is assigned  $a$ .

As is shown in [2, Lemma 3.5 (SOS Cauchy-Schwarz)], this condition is equivalent to the pseudodistribution “satisfying the identity  $Q(X) = b$ ” for  $Q$  of degree up to  $d/2$ .<sup>8</sup>

Intuitively, in this situation degree- $d$  SOS not only “thinks” that it knows a distribution on assignments  $x$  under which  $Q(x)$  has expectation  $b$ , it further thinks that *every* outcome  $x$  in the support of its supposed assignment has  $Q(x) = b$ .

### 3 Random CSPs with Hamming weight constraints

In this section we will prove Theorem 1, which extends the known random CSP lower bounds (as in Theorem 0) to CSPs with a hard Hamming weight constraint on the variable assignment.

#### 3.1 Hypergraph expansion and prior SOS lower bounds for random CSPs

The paper [14] works in the general setting of distributional CSPs with an upper bound of  $K$  on all constraint arities. An instance is thought of as a “factor graph”  $G$ : a bipartite graph with  $n$  variable-vertices,  $m$  constraint-vertices, and edges joining a constraint-vertex to the variable-vertices in its scope. More precisely, the neighborhood  $N(h)$  of each constraint-vertex  $h$  is defined to be an *ordered* tuple of  $k_h$  variable-vertices. We write  $\nu_h$  for the local probability distribution on  $\Omega^{N(h)}$  associated to constraint  $h$ . In [14], each  $\nu_h$  is assumed to be a  $(\tau - 1)$ -wise uniform distribution, where  $\tau$  is a global integer parameter satisfying  $3 \leq \tau \leq K$ . Finally, the graph  $G$  is assumed to satisfy a certain high-expansion condition (discussed in the full version) called the “Plausibility Assumption” involving two parameters  $0 < \zeta < 1$  and  $1 \leq \text{SMALL} \leq n/2$ , assumed to satisfy  $K \leq \zeta \cdot \text{SMALL}$ . In this case, the main theorem of [14] is that there is a SOS-pseudodistribution of degree  $\frac{1}{3}\zeta \cdot \text{SMALL}$  that weakly satisfies all constraints.

In [14] it is assumed that all constraint distributions  $\nu_h$  have the *same* level of uniformity, namely  $(\tau - 1)$ -wise uniformity,  $\tau \geq 3$ . In this work, in order to incorporate Hamming weight constraints on the assignment, we would like to consider the possibility that different constraint distributions have different levels of uniformity. To that end, suppose that each  $\nu_h$  is  $(t_h - 1)$ -wise uniform, where the  $t_h$ ’s are various integers. Slightly more broadly than [14], we allow  $1 \leq t_h \leq k_h + 1$  for all  $h$ , and we allow the constraints to have arity  $k_h$  as low as 1.

In the full version, we examine how these assumptions affect the proofs in [14]. The upshot is Theorem 5 below. Before we give the theorem, we briefly introduce some notation and comments: A “constraint-induced” subgraph  $H$  is a subgraph of the factor graph  $G$  given by choosing some set of constraints  $C$ , as well as all edges and constraint-vertices adjacent to  $C$ . We write  $c(H)$  for the number of constraints in  $H$ ,  $e(H)$  for the number of edges,  $v(H)$  for the number of variable-vertices, and  $T(H) = \sum_{h \in \text{cons}(H)} t_h$ . To reduce to (3.1) in the following theorem, we use the observation in the full version that adding edges to a subgraph to make it constraint-induced can only decrease “income”.

For notational simplicity we have also adjusted the parameters  $\zeta$  and  $\text{SMALL}$  by factors of 2.

<sup>8</sup> In this paper we are flexible when it comes to constant factors in the degree. For this reason we need not worry about this factor-2 loss in the degree, as a degree- $2d$  pseudoexpectation which satisfies an identity with pseudovariance 0 automatically gives a degree- $d$  pseudoexpectation which satisfies the identity exactly.

► **Theorem 5** (Essentially from [14]). *Let  $0 < \zeta < 1$ ,  $\text{SMALL} \leq n$  and assume all constraint-vertices in  $G$  have arity at most  $\zeta \cdot \text{SMALL}$ .*

*Suppose that for every set of nonempty constraint-induced subgraph  $H$  with  $c(H) \leq \text{SMALL}$ , it holds that*

$$v(H) \geq e(H) - \frac{T(H)}{2} + \zeta c(H). \quad (3.1)$$

*Then there is an SOS-pseudodistribution of degree  $\frac{1}{3}\zeta \cdot \text{SMALL}$  that weakly satisfies all constraints.*

There are a lot of parameters in the above theorem, and our goal is not to derive the most general possible quantitative result. Instead we’ll simply work out some of the basic consequences.

A basic setting treated in [14], relevant for Theorem 0, is the following. For a fixed small  $t$  we choose a random CSP with  $n$  variables and  $\Delta n$  constraints, with each constraint supporting a  $(t - 1)$ -wise uniform distribution. E.g., in random 3SAT,  $t = 3$ . Then if

$$\Delta = \text{const} \cdot \left( \frac{n}{\text{SMALL}} \right)^{\frac{t}{2} - 1 - \zeta}$$

for a sufficiently small positive constant, it is shown in [14] that the main condition (3.1) indeed holds with high probability. Choosing, say,  $\zeta = \frac{1}{\log n}$  and  $\text{SMALL} = \text{polylog}(n)$ , we see that, with high probability, we will have weakly satisfying pseudodistributions of degree  $\text{polylog}(n)$  even when  $\Delta = \tilde{\Theta}(n^{t/2-1})$ .

In fact, it’s possible to show that we have such pseudoexpectations when there are, simultaneously,  $n^{1.5}/\text{polylog}(n)$  2-wise-supporting constraints, and  $n^2/\text{polylog}(n)$  3-wise-supporting constraints, and  $n^{2.5}/\text{polylog}(n)$  4-wise-supporting constraints, . . . and also  $n/\text{polylog}(n)$  1-wise-supporting constraints, and  $n^5/\text{polylog}(n)$  0-wise-supporting constraints.

### 3.2 Expansion in the presence of matching and unary constraints

However, if we want to impose a cardinality constraint by way of adding 1-wise independent 2-ary  $\neq$  constraints, then  $n/\text{polylog}(n)$  such constraints will not suffice. Indeed, what we would like to now show is that if the 1-wise-supporting constraints are carefully chosen to not overlap, we can add a full, linear-sized “matching” of them without compromising the lower bound. Then, when  $\Omega = \{0, 1\}$ , we can impose the 1-wise-uniform constraints  $x_1 \oplus x_2 = 1$ ,  $x_3 \oplus x_4 = 1, \dots, x_{n-1} \oplus x_n = 1$  and thereby force the pseudoexpectation to satisfy the global constraint  $\sum_i x_i = \frac{n}{2}$ .

► **Theorem 6.** *Fix a uniformity parameter  $3 \leq t \leq O(1)$ , an arity  $k \leq O(1)$ , a number  $U = O(\sqrt{n})$  of “unary” constraints, and a small failure probability  $0 < p < 1/2$ . Assume also that  $\zeta \leq 1/2$ .*

*Suppose we form a random factor graph with  $n$  variable-vertices and  $\Delta n$  constraint-vertices  $\mathcal{C}$  of arity  $k$ ; assume each constraint-vertex is equipped with an associated  $(t - 1)$ -wise uniform distribution.*

*Furthermore, suppose we add in two sets  $\mathcal{M}_1, \mathcal{M}_2$  of nonrandom, nonoverlapping constraints, whose associated variable vertices partition  $[n]$ . The “unary” constraints of  $\mathcal{M}_1$  should satisfy  $|\mathcal{M}_1| \leq U$  and have an associated 0-wise uniform distribution; the “matching” constraints of  $\mathcal{M}_2$  should be of constant arity and have an associated 1-wise uniform distribution.*

Then provided

$$\Delta \leq \text{const} \cdot p \cdot \left( \frac{n}{\text{SMALL}} \right)^{\frac{t}{2}-1-\zeta'}$$

for a sufficiently small universal constant, and  $\zeta' = (k+1)\zeta$ , the expansion condition in Theorem 5 holds except with probability at most  $p$ .

The proof of Theorem 6, appearing in the full version, uses standard combinatorial techniques for verifying the expansion of random graphs. Due to the fact that the unary and matching constraints are deterministic, we must augment these standard techniques with some straightforward case analysis. In fact, we only prove the theorem under the assumption that the constraints of  $\mathcal{M}_2$  have arity 2; the more general case of constant arity is a slight elaboration that we omit.

### 3.3 Lower bound for CSPs with Hamming weight constraints

As in [14], we observe that for a given  $\Delta \geq 10$ , a good choice for  $\zeta$  is  $\frac{1}{\log \Delta}$ . This yields the following corollary, which we will show implies Theorem 1:

► **Theorem 7.** *Let  $\nu$  be a  $(t-1)$ -wise uniform distribution on  $\{0, 1\}^k$ . Consider a random  $n$ -variable,  $m = \Delta n$ -constraint  $k$ -ary distributional CSP, in which each constraint distribution is  $\nu$  up to a negation pattern in the  $k$  inputs. (All such “reorientations” are still  $(t-1)$ -wise uniform.) Suppose we also impose the following nonrandom distributional constraints:*

- *The 0-wise uniform constraints  $x_1 = b_1, x_2 = b_2, \dots, x_U = b_U$ , for some string  $b \in \{0, 1\}^U$  with  $U = O(\sqrt{n})$ ;*
- *The 1-wise uniform constraint that  $(x_{U+1}, x_{U+2})$  is uniform on  $\{(0, 1), (1, 0)\}$ , and similarly for the pairs  $(x_{U+3}, x_{U+4}), \dots, (x_{n-1}, x_n)$ .*

*Then with high probability, there is an SOS-pseudodistribution of degree  $D = \Omega\left(\frac{n}{\Delta^{2/(t-2)} \log \Delta}\right)$  that weakly satisfies all constraints.*

Let us now see why this implies Theorem 1. In this “ $\beta = 1$ ” scenario, we have a  $(t-1)$ -wise uniform  $\nu$  supported on the satisfying assignments for  $P$ . Whenever the random CSP( $P^\pm$ ) instance has a  $P$ -constraint with a particular literal pattern, we impose the analogous  $\nu$ -constraint with equivalent negation pattern. Now the SOS-pseudodistribution  $\tilde{\mathbb{E}}[\cdot]$  promised by Theorem 6 weakly satisfies all these  $\nu$ -constraints, and hence satisfies all the  $P$ -constraints. Furthermore, it also has  $\tilde{\mathbb{E}}[x_i] = b_i \in \{0, 1\}$  for all  $i \leq U$ , and  $\tilde{\mathbb{E}}[x_i(1-x_{i+1})] + \tilde{\mathbb{E}}[(1-x_i)x_{i+1}] = 1$  for all pairs  $(i, i+1) = (U+1, U+2), \dots, (n-1, n)$ , by weak satisfaction. Notice that the latter implies

$$\tilde{\mathbb{E}}[(x_i + x_{i+1} - 1)^2] = 1 - \tilde{\mathbb{E}}[x_i] - \tilde{\mathbb{E}}[x_{i+1}] + 2\tilde{\mathbb{E}}[x_i x_{i+1}] = 0,$$

and hence the SOS solution satisfies  $x_i + x_{i+1} = 1$  with pseudovariance zero. Similarly (and easier), it satisfies the identity  $x_i = b_i$  for all  $i \leq U$ . It now follows that the pseudodistribution satisfies the identity  $\sum_{i=1}^n x_i = \frac{n}{2} + (|b| - \frac{U}{2})$  with pseudovariance zero, and this completes the proof of Theorem 1, because we can take any  $|b| \in \{0, \dots, U\}$ .

## 4 Exact Local Distributions on Composite Predicates

In this section and in Section 5 we will show how to satisfy the constraint  $\text{OBJ}(x) = \beta$  exactly, with pseudovariance zero. Our strategy will be to group predicates together into “composite” predicates, and then prove that there is a local  $(t-1)$ -wise uniform distribution

which is moreover supported on variable assignments for which an exact  $\beta$ -fraction of the predicates within the composite predicate are satisfied. We'll then apply Theorem 6 to the composite predicates.

We begin with an easier proof for the case when there is a pairwise-uniform distribution over satisfying assignments to our predicate in Section 4.1, and later in Section 4.2 we handle  $t$ -wise uniform distributions for larger  $t$ . While the pairwise-uniform theorem is less general, the proof is simpler and it already suffices for all of our bisection applications.

#### 4.1 Pairwise-uniform distributions over $\beta$ -satisfying assignments

Recall our setting: we have a Boolean  $k$ -ary predicate  $P : \{\pm 1\}^k \rightarrow \{0, 1\}$  and a pairwise-uniform distribution  $\nu$  over assignments  $x \in \{\pm 1\}^k$  such that  $\mathbb{E}[P(x)] = \beta$ . The following theorem states that we can extend  $\nu$  into a distribution  $\theta$  over assignments to groups of  $r$  predicates at a time,  $\{\pm 1\}^{r \times k}$  so that exactly  $(\beta - \varepsilon) \cdot r$  of the  $r$  predicates are satisfied by any assignment  $y \sim \theta$ .

► **Theorem 8.** *Let  $P : \{\pm 1\}^k \rightarrow \{0, 1\}$  be a  $k$ -ary Boolean predicate, and let  $\nu$  be a pairwise-uniform distribution over assignments  $\{\pm 1\}^k$  with the property that  $\nu(x)$  is rational for each  $x \in \{\pm 1\}^k$ ; that is, there exist a multiset  $S \subseteq \{\pm 1\}^k$  such that for each  $x \in \{\pm 1\}^k$ ,  $\nu(x) = \mathbb{P}_{s \sim S}(s = x)$ . Suppose also that  $\mathbb{E}_{x \sim \nu}[P(x)] = \beta$ , and that this is more than the expectation under the uniform distribution, so  $\beta > \mathbb{E}_{x \sim \{\pm 1\}^k}[P(x)]$ .*

*Then for any constant  $\varepsilon > 0$ , there exists an integer  $r = O_{\varepsilon, k}(|S|^3)$  and a rational  $\tilde{\varepsilon} \leq \varepsilon$  so that there is a pairwise-uniform distribution  $\theta$  over assignments to groups of  $r$  predicates,  $\{\pm 1\}^{r \times k}$  such that exactly  $(\beta - \tilde{\varepsilon})r$  of the predicates are satisfied by any assignment  $y \sim \theta$ .*

Throughout we'll refer to the assignments in the support of  $\theta$  as *matrices*, with each row of the  $r \times k$  matrix corresponding to the assignment for a single copy of the predicate.

Since we have assumed that the probability of seeing any string in the support of  $\nu$  is rational, without loss of generality we can assume that  $\nu$  is uniform over some multiset  $S \subseteq \{\pm 1\}^k$ . As a first guess at  $\theta$ , one might try to take  $r = c \cdot |S|$  for some positive integer  $c$ , make  $c$  copies of the multiset  $S$ , and use a random permutation of the elements of this multiset to fill the rows of an  $r \times k$  matrix. But this distribution is not *quite* pairwise uniform. The issue is that because each individual bit is uniformly distributed, every column of the matrix will always be perfectly balanced between  $\pm 1$ . Therefore the expected product of two distinct bits in a given column is

$$\frac{1}{\binom{c|S|}{2}} \left( \binom{\frac{1}{2}c|S|}{2} (-1)^2 + \binom{\frac{1}{2}c|S|}{2} (+1)^2 + \binom{\frac{1}{2}c|S|}{2} (+1)(-1) \right) = -\frac{1}{c|S| - 1} \neq 0.$$

So, the bits within a particular column have a slight negative correlation.

We'll compensate for this shortcoming as follows: we will randomly choose an element  $s$  in the support of  $\nu$  to repeat multiple times. This may in turn alter the number of predicates satisfied out of the  $r$  copies of  $P$ , whereas our express goal was to satisfy the exact same number of predicates under every assignment. To adjust for this, we'll mix in some rows from the uniform distribution over  $\{\pm 1\}^k$ , where the number of rows we mix in will depend on whether  $P(s) = 1$  or  $0$ .

**Proof.** Let  $S$  be a multiset of strings in  $\{\pm 1\}^k$  such that  $\mathbb{P}_{s \sim S}(s = a) = \nu(a)$ . We will also require a multiset  $T \subseteq \{\pm 1\}^k$  which is a well-chosen mixture of  $\nu$  and the uniform distribution; the following claim shows that we can choose such a set. Here, we take some care in choosing this combination; the exact choice of parameters will not matter until later.

► **Claim 9.** For any constant  $\varepsilon > 0$ , there is a constant  $L = O(1/\varepsilon)$  and a constant  $R \geq 1$  so that there are multisets  $S', T \subseteq \{\pm 1\}^k$  with the following properties:  $S'$  is  $RL2^k$  copies of  $S$ ,  $T$  has size  $|T| = |S'| = LR2^k|S|$ , and  $\mathbb{P}_{x \sim T}[P(x) = 1] = \beta - \varepsilon'$ , where  $\varepsilon > \varepsilon' \stackrel{\text{def}}{=} \frac{1}{L}(\beta - \mathbb{E}_{x \sim \{\pm 1\}^k}[P(x)])$ .

**Proof.** Let  $s = |S|$ , and let  $U = \{\pm 1\}^k$ . Suppose that  $\eta 2^k$  of  $U$ 's assignments satisfy  $P$ . Define  $T$  to be the multiset given by  $2^k(L-1)R$  copies of  $S$  and  $sR$  copies of  $U$ . We have

$$\mathbb{P}_{x \sim T}[P(x) = 1] = \frac{1}{2^k L R s} (\beta s \cdot 2^k (L-1)R + \eta 2^k \cdot sR) = \beta - \frac{\beta - \eta}{L}.$$

By choosing  $L$  large as a function of  $\varepsilon$ , we can make this probability as small as we want. ◀

For convenience, let  $\ell = 2^k L R |S|$ . Set the number of rows  $r = d\ell$  for an integer  $d = O_\varepsilon(\ell^2)$  to be specified later. We also let  $a, b_1, b_0, c_1, c_0$  be integers which will specify the number of rows from  $S', T$ , and the repeated assignment set; we'll set the integers later, but we will require the property that

$$d = a + b_1 + c_1 = a + b_0 + c_0. \quad (4.1)$$

We generate a sample from  $\theta$  in the following fashion:

1. Sample  $s \sim S$ , and fill the first  $a\ell$  rows with copies of  $s$ . Call these the  $A$  rows.
2. Set  $i = P(s)$ , that is  $i = 1$  if  $s$  satisfies  $P$  and  $i = 0$  otherwise.
3. Fill the next  $b_i\ell$  rows with  $b_i$  copies of each string in  $S'$ . Call these the  $B$  rows.
4. Fill the last  $c_i\ell$  rows with  $c_i$  copies of each string in  $T$ . Call these the  $C$  rows.
5. Randomly permute the rows of the matrix.

If  $\delta\ell$  assignments in  $T$  are satisfying and  $\beta\ell$  assignments in  $S'$  are satisfying, to ensure that the number of satisfying rows is always the same we enforce the constraint

$$a\ell + b_1\beta\ell + c_1\delta\ell = b_0\beta\ell + c_0\delta\ell, \quad (4.2)$$

Now we handle uniformity. We will prove that all of the degree-1 and degree-2 moments of the bits in the matrix are uniform under  $\theta$ . First, we argue that the degree-1 moments are zero, and that the correlation of any two bits in the same row is zero.

► **Claim 10.** The bits in a single row of  $M$  are pairwise uniform.

**Proof.** We can condition on the row type,  $A, B$ , or  $C$ . For each type of row, there is a multiset  $U$  such that  $\mathbb{E}[M_{ij}] = \mathbb{E}_{x \sim U}[x_j] = 0$  by the pairwise uniformity of the uniform distribution over  $U$ . The same argument proves the statement for the product of two bits in a fixed row. ◀

Thus, it suffices to prove that the bits in each column are pairwise-uniform; this is because the pairwise uniformity of rows implies that we can fix the values of any entire column, and the remaining individual bits in other columns will remain uniformly distributed. So we turn to proving that the columns are pairwise-uniform.

► **Claim 11.** If we choose  $a, b_0, b_1, c_0, c_1$  so that

$$a(\ell - 1) - \beta(b_1 + c_1) - (1 - \beta)(b_0 + c_0) = 0,$$

then the bits in a single row of  $M$  are pairwise uniform.



**Proof.** We'll prove this by computing the expected product of two distinct bits,  $x$  and  $y$ , which both come from the  $i$ th column of  $M$ . We will compute the conditional expectation of  $xy$  given the group of rows that  $x, y$  were sampled from.

We first notice that conditioned on  $x$  coming from one type of row and  $y$  coming from another,  $x$  and  $y$  are independent of each other, and by the uniformity of individual bits in each group,  $\mathbb{E}[xy \mid x, y \sim \text{different groups}] = 0$ .

Restricting our attention now to pairs of bits from within the same group, we compute the conditional expectations. If both bits come from the  $A$  rows, they are perfectly correlated. On the other hand, if both bits come from the  $B$  or  $C$  rows their correlation is as we computed above,  $-\frac{1}{b\ell-1}$  or  $-\frac{1}{c\ell-1}$  respectively. Therefore we can simplify

$$\begin{aligned}
 \mathbb{E}[xy] &= \beta \cdot \mathbb{E}[xy \mid P(a) = 1] + (1 - \beta) \cdot \mathbb{E}[xy \mid P(a) = 0] \\
 &= +\beta \left( \frac{\binom{a\ell}{2}}{\binom{d\ell}{2}} + \mathbb{E}[xy \mid x, y \sim B, P(a) = 1] \cdot \frac{\binom{b_1\ell}{2}}{\binom{d\ell}{2}} + \mathbb{E}[xy \mid x, y \sim C, P(a) = 1] \cdot \frac{\binom{c_1\ell}{2}}{\binom{d\ell}{2}} \right) \\
 &\quad + (1 - \beta) \left( \frac{\binom{a\ell}{2}}{\binom{d\ell}{2}} + \mathbb{E}[xy \mid x, y \sim B, P(a) = 0] \cdot \frac{\binom{b_0\ell}{2}}{\binom{d\ell}{2}} + \mathbb{E}[xy \mid x, y \sim C, P(a) = 0] \cdot \frac{\binom{c_0\ell}{2}}{\binom{d\ell}{2}} \right) \\
 &= \frac{\ell}{2\binom{d\ell}{2}} \left( a(a\ell - 1) - \beta(b_1 + c_1) - (1 - \beta)(b_0 + c_0) \right), \tag{4.3}
 \end{aligned}$$

where (4.3) gives us the condition of the claim. ◀

Finally, we are done given that we can find positive integers satisfying the constraints

$$\begin{aligned}
 d - a &= b_1 + c_1 = b_0 + c_0 && \text{(from (4.1))} \\
 0 &= a + \beta(b_1 - b_0) + \delta(c_1 - c_0) && \text{(from (4.2))} \\
 0 &= a(\ell - 1) - \beta(b_1 + c_1) - (1 - \beta)(b_0 + c_0) && \text{(from (4.3))}
 \end{aligned}$$

The following can be verified to satisfy the constraints above:

$$\begin{aligned}
 a &:= 2(\beta - \delta)\ell; && b_1, c_0 := ((\beta - \delta)(\ell - 1) - 1)\ell; \\
 b_0, c_1 &:= ((\beta - \delta)(\ell - 1) + 1)\ell; && d := 2(\beta - \delta)\ell^2.
 \end{aligned}$$

By our choice of  $\ell = 2^k |S|LR$  and since  $\beta - \delta = \frac{\beta - \mathbb{E}[P(x)]}{L}$ , we can choose  $R$  large enough so that  $(\beta - \delta)(\ell - 1) > 1$ , and because  $\beta\ell$  and  $\delta\ell$  are integers, these are all also positive integers, as required.

We compute the number of satisfied assignments as a function of the total, which is

$$\beta \frac{b_0}{d} + \delta \frac{c_0}{d} = \beta - \frac{\varepsilon'}{2} + \frac{1}{\ell} \left( \frac{1 + \varepsilon'}{2} - \beta \right).$$

The conclusion thus holds, with  $\tilde{\varepsilon} \stackrel{\text{def}}{=} \frac{\varepsilon'}{2} - \frac{1}{\ell} \left( \frac{1 + \varepsilon'}{2} - \beta \right)$ . ◀

## 4.2 $t - 1$ -wise uniform distributions over $\beta$ -satisfying assignments

We now prove the generalization of the statement in the previous section to  $t$ -wise uniform distributions over  $\beta$ -satisfying assignments.

► **Theorem 12.** *Let  $P : \{\pm 1\}^k \rightarrow \{0, 1\}$  be a  $k$ -ary Boolean predicate, let  $t \geq 2$  be an integer, and let  $\nu$  be a  $(t-1)$ -wise uniform distribution over assignments  $\{\pm 1\}^k$  so that there exist a multiset  $S \subseteq \{\pm 1\}^k$  such that for each  $x \in \{\pm 1\}^k$ ,  $\nu(x) = \mathbb{P}_{s \sim S}(s = x)$ . Suppose also that  $\mathbb{E}_{x \sim \nu}[P(x)] = \beta > \mathbb{E}_{x \sim \{\pm 1\}^k}[P(x)]$ .*

*Then for any constant  $\varepsilon > 0$ , there exists an integer  $r = O_{\varepsilon, k}(|S|^4)$  and a rational  $\tilde{\varepsilon} \leq \varepsilon$  so that there is a  $(t-1)$ -wise uniform distribution  $\theta$  over assignments to groups of  $r$  predicates,  $\{\pm 1\}^{r \times k}$  such that exactly  $(\beta - \tilde{\varepsilon})r$  of the predicates are satisfied by any assignment  $y \sim \theta$ .*

The proof will use a similar, though slightly more involved, construction of  $\theta$  than in the pairwise case. It may be helpful to note that the choice of  $t = 3$  in Theorem 12 will not give the same construction as in Theorem 8 (although of course one could set  $t = 3$  and obtain a result for pairwise-uniform  $\nu$ ). In particular, it will not be enough to choose one string to repeat many times in order to improve the column-wise correlations. Instead, we will repair the correlations in one column at a time, by sampling some subset of the bits in each column from a bespoke distribution, designed to make the columns  $(t-1)$ -wise independent. We will have to be careful with the choice of distribution, so that we can still control the number of satisfying assignments in  $M$  as a whole.

**Proof.** As in the proof of Theorem 8, we will require a well-chosen convex combination of  $\nu$  and the uniform distribution to ensure that the number of satisfying assignments is always the same. We appeal to Claim 9, taking  $S'$  and  $T$  to be as described there, with  $L = O(1/\varepsilon\psi)$  (to be set more precisely later) and  $R = 1$ . For convenience let's let  $\ell \stackrel{\text{def}}{=} 2^k L |S|$  and let's let  $\delta = \beta - \varepsilon'$ .

We also call  $S'_{i=1}$  and  $S'_{i=-1}$  to be the sub-multisets of  $S'$  which have the  $i$ th bit set to 1 and  $-1$  respectively. We notice that a uniform sample from  $S'_{i=1}$  is equivalent to a uniform sample from  $\nu$  conditioned on the  $i$ th bit being 1. Also by the  $(t-1)$ -wise uniformity we have  $|S'_{i=1}| = \ell/2$ . Notice that since  $S'$  is made up of  $2^k L$  copies of  $S$ , the discrepancy in the number of satisfying assignments between  $S_{i=1}$  and  $S_{i=-1}$  is always an integer multiple of  $2^k L$ .

Set  $r$ , the number of rows, be an integer which we will specify later. We also choose the integer  $a$  to represent the size of the correction rows, and  $b_n, c_n$ , the number of copies of  $S'$  and  $T$  for each  $n \in [ak\ell/(2^k L)]$  (where  $n2^k L$  is the number of satisfying assignments in the correction rows). To make sure the number of rows always adds up to  $r$ , we'll need the constraint,

$$r = ak\ell + b_n\ell + c_n\ell \quad \forall z \in \{\pm 1\}^k \quad (4.4)$$

In order to make sure that the columns are  $(t-1)$ -wise independent, we require a ‘‘column repair’’ distribution  $\kappa$  over  $\{\pm 1\}^{a\ell}$ . We will specify this distribution later; for now, we need only that  $\kappa$  is symmetric and that the number of 1s in any  $z \sim \kappa$  is a multiple of  $\ell/2$ . The latter property is because, when we choose some part of column  $i$  according to  $\kappa$ , we will want to fix the rows with copies of  $S_{i=1}$  and  $S_{i=-1}$ .

We generate a sample  $M \in \{\pm 1\}^{r \times k}$  from  $\theta$  in the following fashion:

1. For each  $i \in [k]$ , independently sample a string  $z_i \sim \kappa$ . Add  $a\ell$  rows to  $M$ , where in the  $i$ th column we put the bits of  $z_i$ , and we set the remaining row bits so that if  $z_i$  has  $(a - a')\ell/2$  entries of value 1 and  $a'\ell/2$  entries of value  $-1$ , then we end up with  $a'$  copies of  $S'_{i=-1}$  and  $a - a'$  copies of  $S'_{i=1}$ . Call these rows  $A_i$ .
2. Compute the integer  $n$  such that  $n \cdot 2^k L$  is the number of rows in  $\cup_{i=1}^k A_i$  containing satisfying assignments to  $P$ , given our choices of  $z_i \forall i \in [k]$ .

3. Add  $b_n \ell$  rows to  $M$  which contain  $b_n$  copies of each string from  $S$ . Call these rows  $B$ .
4. Add  $c_n \ell$  rows to  $M$  which contain  $c_n$  copies of each string from  $T$ . Call these rows  $C$ .
5. Randomly permute the rows of  $M$ .

So that the number of satisfying assignments  $\Lambda$  is always the same, we require that

$$\Lambda = n2^k L + b_n \cdot \beta \ell + c_n \cdot \delta \ell \quad \forall n \in [ak\ell/(2^k L)] \quad (4.5)$$

Now, we will derive the conditions under which  $(t-1)$ -wise independence holds. As above, we first consider bits that are all contained in a fixed row.

► **Claim 13.** *The bits in a single row of  $M$  are  $(t-1)$ -wise uniform.*

**Proof.** We can condition on the row type,  $A_1, \dots, A_k, B$ , or  $C$ . Sampling a uniform row from  $B$  or  $C$  is equivalent to sampling from  $\nu$  or  $\gamma$ , which are  $(t-1)$ -wise uniform. Since  $\kappa$  is symmetric, sampling a row from  $A_i$  is equivalent to sampling from  $\nu$  as well, and we are done. ◀

From the claim above, if we condition on the value in  $d < t-2$  columns, the remaining  $t-2-d$  columns will remain identically distributed; this is because the rows are  $(t-1)$ -wise uniform, so after conditioning the distribution in each row will remain  $(t-1-d)$ -wise uniform. Thus, proving that each column is  $(t-1)$ -wise uniform suffices to prove  $(t-1)$ -wise uniformity on the whole.

The following lemma states that we may in fact choose  $\kappa$  so that this condition holds exactly.

► **Lemma 14.** *Let  $y \in \{\pm 1\}^{r-al}$  be a perfectly balanced string. If  $al > h_1 \cdot \sqrt{tr}$  for a fixed constant  $h_1$  and  $\sqrt{r} \geq (t-1)\ell 2^{h_2 t}$  for a fixed constant  $h_2$ , then there is a distribution  $\kappa$  over  $\{\pm 1\}^{al}$ , supported on strings which have a number of 1s which is a multiple of  $\ell/2$ , such that if  $x$  is sampled by choosing  $z \sim \kappa$ , concatenating  $z$  with  $y$  and applying a random permutation, then for any  $S \subset [r]$  with  $|S| \leq t-1$ ,  $\mathbb{E}[x^S] = 0$ .*

Since each column is distributed as the string  $x$  described in the lemma statement, the lemma suffices to give us  $(t-1)$ -wise uniformity of the columns. We'll prove the lemma below, but first we conclude the proof of the theorem statement.

We now choose the parameters to satisfy our constraints. We have the requirements:

$$\begin{aligned} \Lambda &= n2^k L + b_n \beta \ell + c_n \delta \ell \quad \forall n \in [ak\ell/(2^k L)] && \text{(from (4.5))} \\ r - ak\ell &= b_n \ell + c_n \ell \quad \forall n \in [ak\ell/(2^k L)] && \text{(from (4.4))} \\ al &\geq h_1 \sqrt{tr} && \text{(from Lemma 14)} \\ \sqrt{r} &\geq (t-1)\ell 2^{h_2 t} && \text{(from Lemma 14)} \end{aligned}$$

where  $h_1$  and  $h_2$  are universal constants. The below choice of integer parameters satisfies these requirements, as well as the requirement of always being non-negative:

$$\begin{aligned} u &= \left( \beta - \mathbb{E}_{x \sim \{\pm 1\}^k} [P(x)] \right) 2^k |S|; & L &= u \cdot \max(1, \lceil h_1 + h_2 \rceil) \cdot \left\lceil \frac{1}{\varepsilon} \right\rceil \cdot k; & \ell &= 2^k L |S|; \\ a &= \left\lceil h_1 2^{h_2 t/2} t k \right\rceil; & r &= \left\lceil \frac{a^2}{h_1^2 t} \right\rceil \cdot \ell^2; \\ b_0 &= \frac{1}{2} \left( \frac{1}{\ell} r - ak \right); & b_{n+1} &= b_n - \frac{2^k L}{u}; \end{aligned}$$

$$c_0 = \frac{1}{2} \left( \frac{1}{\ell} r - ak \right); \quad c_{n+1} = c_n + \frac{2^k L}{u}.$$

Finally, we have that the fraction of satisfying rows in  $M$  is always exactly

$$\frac{\Lambda}{r} = \frac{\frac{1}{2}(r - ak\ell)\beta + \frac{1}{2}(r - ak\ell)\delta}{r} = \beta - \frac{\varepsilon'}{2} - O\left(\frac{ak\ell}{r}\right).$$

The latter term is  $O(\frac{1}{\ell})$ , and we have chosen  $L$  large enough so that it is smaller than  $\varepsilon'/2$ .  $\blacktriangleleft$

**Proof of Lemma 14.** For convenience, call  $m \stackrel{\text{def}}{=} r - a\ell$ . Recall that we take  $x$  to be sampled by taking a balanced string  $y \in \{\pm 1\}^m$ , sampling  $z \sim \kappa$ , appending  $z$  to  $y$  and then applying a uniform permutation to the coordinates.

We will solve for  $\kappa$  with a linear program (LP) over the probability  $p_z$  of each string  $z \in \{\pm 1\}^{a\ell}$ . We have the program

$$\begin{aligned} \forall S \in [m], |S| \in \{1, \dots, t-1\} : & \sum_{\substack{z \in \{\pm 1\}^{a\ell} \\ (\ell/2) \mid \sum_j z_j}} \mathbb{E} \left[ x^S \mid \sum_i x_i = \sum_{j \in [a\ell]} z_j \right] \cdot p_z = 0 \\ \forall z \in \{\pm 1\}^{a\ell} \text{ s.t. } \frac{\ell}{2} \mid \sum_j z_j : & p_z \geq 0 \end{aligned}$$

Since we can take any solution to this LP and scale the  $p_z$  so that they sum to 1, the feasibility of this program implies our conclusion. So suppose by way of contradiction that this LP is infeasible. Then Farkas' lemma implies that there exists a  $q \in \mathbb{R}^{t-1}$  such that

$$\forall z \in \{\pm 1\}^{a\ell} \text{ s.t. } \frac{\ell}{2} \mid \sum_j z_j, \quad \sum_{\substack{S \subseteq [m] \\ |S| \in \{1, \dots, t-1\}}} \mathbb{E} \left[ x^S \mid \sum_i x_i = \sum_{j \in [a\ell]} z_j \right] \cdot y_S > 0.$$

Without loss of generality, we scale  $q$  so that  $\sum_S q_S^2 = 1$ . Moreover by the symmetry of the expectation over subsets  $S$ , we can assume that  $q_S = q_T$  whenever  $|S| = |T|$ . This implies that the degree- $t$  mean-zero polynomial

$$q(x) = \sum_{\substack{S \subseteq [m] \\ 1 \leq |S| \leq t-1}} q_S \cdot \chi_S(x)$$

has positive expectation over every layer of the hypercube with  $|\sum_i x_i| = d$  such that  $d \leq a\ell$  and  $\frac{\ell}{2} \mid d$ . Furthermore,  $q$  is a symmetric polynomial, which implies that it takes the same value on all inputs of a fixed Hamming weight; this implies that it takes positive values on every inputs  $x$  with  $|\sum x_i| \in [2a] \cdot \frac{\ell}{2}$ .

The following fact will give us the contradiction we desire:

► **Fact 15** (Tails of low-degree polynomials [6], see Theorem 4.1 in [1]). *Let  $f : \{0, 1\}^m \rightarrow \mathbb{R}$  be a degree- $t$  polynomial with mean zero and variance 1. Then, there exist universal constants  $c_1, c_2 > 0$  such that  $\mathbb{P}[p \leq -2^{-c_1 t}] \geq 2^{-c_2 t}$ .*

We will show that since  $q$  takes positive value on every hypercube slice of discrepancy  $\frac{\ell}{2} \cdot [2a]$ , this implies that it takes positive values on most hypercube slices with discrepancy at most  $a\ell$ . Because we have chosen  $a\ell$  so that this comprises the bulk of the hypercube, this in turn will contradict Fact 15.

In fact, because  $q$  is symmetric and of degree  $t - 1$  over the hypercube, we can equivalently write  $q$  as a degree- $(t - 1)$  polynomial in the single variable  $x' := \sum_i x_i$ ,  $q(x) = g(\sum_i x_i) = \sum_{s \in \{0, \dots, t-1\}} g_s \cdot (\sum_i x_i)^s$ . Viewing  $g$  as a univariate polynomial over the reals,  $g$  has at most  $t - 1$  roots. Therefore we conclude that  $q$  can only be non-positive on at most  $t - 1$  intervals of layers of discrepancy  $(i\ell/2, (i + 1)\ell/2)$ . So for at least  $2a\ell - (t - 1)\frac{\ell}{2}$  slices of the hypercube around 0,  $q$  takes positive value.

Each slice of the hypercube has probability mass at most  $\frac{1}{\sqrt{\frac{\pi}{2}(m+a\ell)}}$ . By our choice of  $a, \ell, m$  and by a Chernoff bound,

$$\mathbb{P}(q(x) > 0) \geq \mathbb{P}\left(\left|\sum_i x_i\right| \leq a\ell\right) - \frac{(t-1)\ell}{2\sqrt{m+a\ell}} > 1 - 2^{-c_2 t},$$

which contradicts Fact 15. ◀

## 5 SOS lower bounds for CSPs with exact objective constraints

In this section we put things together and show how to extend Theorem 12 to prove Theorem 2. As discussed briefly in Section 1.1, our random instance of the  $CSP(P^\pm)$  will be sampled in a somewhat non-standard way, which we will refer to as “batch-sampling.” This is because, in order to apply Theorem 12 to a random instance  $\Phi$  of a Boolean CSP, we need to partition  $\Phi$ ’s constraints into groups of  $r$  non-intersecting constraints for some integer  $r$ , while also maintaining the expansion properties required by Theorem 5.

We first prove Theorem 2 as stated, for random CSPs sampled from a slightly different distribution. Then in Section 5.2 we show that for a “standard” random CSP with  $m = o(n^{3/2})$  constraints, we can still get a theorem along the lines of Theorem 2.

### 5.1 Exact objective constraints for batch-sampled random CSPs

Suppose that  $P$  is a  $k$ -ary predicate, and let  $r$  be some positive integer which divides  $m$ . We’ll “batch-sample” an  $n$ -variate random  $CSP(P^\pm)$  with  $m$  clauses as follows:

1. Choose independently  $m/r$  subsets each of  $r \cdot k$  distinct variables uniformly at random from  $[n]$ ,  $S_1, \dots, S_m$
2. For each  $j \in [m/r]$ ,  $S_j = \{x_{i_1}, \dots, x_{i_{rk}}\}$ :
  - Choose a random signing of  $P$ ,  $z_j \in \{\pm 1\}^k$
  - To each block of  $k$  variables in  $S_j$ ,  $(x_{i_{(\ell-1) \cdot k+1}}, \dots, x_{i_{\ell k}})$  for  $\ell \in [r]$ , add the predicate  $P$  with signing  $z_j$ .

► **Theorem 16** (Restatement of Theorem 2). *Let  $P$  be a  $k$ -ary predicate, and let  $\nu$  be a  $(t - 1)$ -wise uniform distribution over  $\{\pm 1\}^k$  under which  $\mathbb{E}_\nu[P] = \beta$ . Then for each constant  $\varepsilon > 0$  there is a choice of positive integer  $r$  such that for a random instance of  $CSP(P^\pm)$  on  $n$  variables with  $m = \Delta n$  constraints for sufficiently large  $\Delta$  and  $r \mid m$ , sampled as detailed above, there is a degree- $\Omega(\frac{n}{\Delta^{2/(t-2)} \log \Delta})$  SOS pseudodistribution which satisfies with pseudovariance zero the constraint  $\text{OBJ}(x) = \beta - \varepsilon_r$ , where  $\varepsilon_r < \varepsilon$ . This is also true when cardinality constraints are imposed as in Theorem 1.*

**Proof.** This distribution over instances is equivalent to the standard notion of sampling a random CSP with  $m/r$  constraints in the composite predicates from Theorem 12: a scope is chosen independently and uniformly at random for each predicate. Therefore, if we replace each collection of constraints corresponding to  $S_j$  with the composite predicate from Theorem 12, and modify  $\nu$  in accordance with the signing  $z_j$ , we have a  $(t - 1)$ -wise uniform

distribution over solutions to the composite predicates supported entirely on assignments which satisfy exactly  $\beta - \varepsilon_r$  of the clauses. Combining this with the expansion theorem (Theorem 6), we have our conclusion. ◀

## 5.2 Exact objective constraints for sparse random CSPs.

Though the batch-sampled distribution over CSPs for which Theorem 2 holds is slightly non-standard, here we show that with minimal effort, we can prove a similar theorem for sparse random instances sampled in the usual manner, when  $m = o(n^{3/2})$ .

► **Theorem 17.** *Let  $P$  be a  $k$ -ary predicate, let  $\nu$  be a  $(t - 1)$ -wise uniform distribution over  $\{\pm 1\}^k$  such that  $\mathbb{E}_\nu[P] = \beta$ , and suppose we sample a random instance  $\Phi$  of a CSP( $P^\pm$ ) in the usual way, by selecting  $m$  random signed  $P$ -constraints on  $n$  variables. Then if  $m = \Delta n = o(n^{3/2})$  for sufficiently large  $\Delta$ , with high probability over the choice of  $\Phi$ , for each  $\varepsilon > 0$  there exists some constant  $\varepsilon_\Phi \leq \varepsilon$  such that there is a degree- $\Omega_\varepsilon(\frac{n}{\Delta^{2/(t-2)} \log \Delta})$  pseudodistribution which satisfies with pseudovariance zero the constraint  $\text{OBJ}(x) = \beta - \varepsilon_\Phi$  and a Hamming weight constraint  $\sum_{i \in [n]} x_i = B$  for  $|B| = O(\sqrt{n})$ .*

**Proof.** Fix  $\varepsilon$ , and let  $r$  be the corresponding constant required to achieve objective  $\text{OBJ}(x) = \beta - \varepsilon^*$  under Theorem 12 for  $\varepsilon^* < \varepsilon/2$ .

We first couple the standard sampling procedure for a random  $P$ -CSP to a different sampling procedure. For simplicity we at first ignore the possible signings of  $P$ , and assume we work only with un-negated variables; later we explain how to modify the proof to accomodate negative literals.

We sample a random CSP by independently and uniformly choosing  $m$  random scopes  $S_1, \dots, S_m$ . For each  $\ell \in \{0, 1, \dots, \lfloor m/r \rfloor - 1\}$ , the probability that  $S_{\ell r + 1}, \dots, S_{(\ell + 1)r}$  have non-intersecting scopes is at least

$$\mathbb{P}[\cap_{j \in [r]} S_j = \emptyset] = \prod_{i=2}^r \mathbb{P}[S_i \cap (\cap_{j < i} S_j) = \emptyset \mid \cap_{j < i} S_j = \emptyset] = \prod_{i=2}^r \left(1 - i \frac{k}{n}\right) \geq 1 - O\left(\frac{r \cdot k}{n}\right).$$

So with high probability for all but  $O(\frac{m}{n})$  of the intervals of constraints  $j \in [\ell \cdot r + 1, (\ell + 1)r]$ , the constraints will be non-intersecting. Call this the “non-intersecting configuration”.

Define a “collision configuration” to be a choice of scopes for which the above condition does not hold; that is, a specific way in which  $S_j$  intersects with one or more  $S_{j'}$  when  $j, j' \in [\ell \cdot r + 1, (\ell + 1)r]$ . Each of the  $\approx \binom{2kr}{r}$  collision configurations has a fixed probability of occurring (which may be easily calculated), and the total sum of these probabilities is at most  $O(r \cdot k/n)$ .

Let  $\mathcal{D}_r^{(m)}$  be the multinomial distribution which describes the number of occurrences of each configuration for a random CSP with  $m$  constraints ( $\lfloor m/r \rfloor$  configurations). We couple the standard sampling procedure with the following alternative sampling procedure: we first sample  $c \sim \mathcal{D}_r^{(m)}$  to determine how many configurations of each type there are. Then, for each collision configuration specified by  $c$ , sample the scope (of size  $< k \cdot r$ ) for each of the collision configurations independently and uniformly at random. Also, sample an additional  $(m \bmod r)$  scopes of  $k$  variables for the “leftover copies” of  $P$ . Finally, sample the scopes of the non-intersecting configurations specified by  $c$  independently uniformly at random. The coupling of the two processes is immediate.

Let  $C$  be the number of collision configurations plus  $(m \bmod r)$ , the number of leftover copies. As shown above, with high probability over  $c \sim \mathcal{D}_r^{(m)}$ , the number of collision configurations is at most  $O(m/n) = o(n^{1/2})$ , so  $C = o(n^{1/2}) = o(m)$ .

From our alternate sampling procedure, we conclude that with high probability we can meet the conditions of Theorem 6 by fixing an arbitrary variable assignment to any collision configuration. That is, we could alternately first sample the collision configurations and leftover copies, and then set all of the variables present inside be set to (say) False. We take note of how many constraints in the  $P$ -CSP are and are not satisfied by this unary assignment, and we correspondingly amend  $\varepsilon^*$  to  $\varepsilon_\Phi$ . Since with high probability at most  $o(m)$  constraints are fixed, we retain the property that  $\varepsilon_\Phi \leq \varepsilon^* + o(1) \leq \varepsilon$ .

Now, if we wish to satisfy a Hamming weight constraint, we add arbitrary matching and unary constraints to get the desired Hamming weight; at most  $O(C)$  unary constraints are needed to compensate for the  $\leq Ckr$  variables we set to False.

Finally, we sample the remaining non-intersecting configurations independently; by Theorem 6 when  $C = o(n^{1/2})$ , the expansion properties we require are met for the composite predicates on the non-intersecting configurations. Since this occurs with high probability, we are done.

To extend the argument to allow predicates on negative literals, we couple with a slightly more elaborate sampling procedure: for each signing pattern  $z \in \{\pm 1\}^k$ , we draw a separate set of  $m_z$  predicates (where  $m_z$  may either be deterministic or sampled from a multinomial distribution). For each signing separately we repeat the argument above, and then in the final sampling procedure we sample counts  $c_z$  for each signing  $z$ , add the leftover copies and collision configurations separately for each signing, add the unary constraints, and then sample the remaining non-intersecting copies. ◀

## 6 Conclusions

In this work we have shown that, in the context of random Boolean CSPs, the following strategies do *not* give SOS any additional refutation power: (i) trying out all possible Hamming weights for the solution; (ii) trying out all possible (exact) values for the objective function. We also gave the first known SOS lower bounds for the Min- and Max-Bisection problems.

We end by mentioning some open directions. There are two technical challenges arising in our work that look approachable. The first is to extend our results from Section 4 on “exactifying” distributions to the case of larger alphabets. The second is to prove (or disprove) that the “random\*” and “purely random” distributions discussed in Remark 1.1 are  $o(1)$ -close (depending on  $m(n)$ ).

Finally, we suggest investigating further strategies for handling hard constraints in the context of SOS lower bounds. Sometimes this is not too difficult, especially when reducing from linear predicates such as 3XOR, where there are perfectly satisfying SOS solutions. Other times, it’s of moderate difficulty, perhaps as in this paper’s main Theorem 1 and Theorem 2. In still other cases it appears to be very challenging.

One difficult case seems to be in the context of SOS lower bounds for refuting the existence of large cliques in random graphs. In [4] it is shown that in a  $G(n, 1/2)$  random graph, with high probability degree- $\Omega(\log n)$  SOS thinks there is a clique of size  $\omega := n^{1/2-\varepsilon}$ . (Here  $\varepsilon > 0$  can be any constant.) However, it’s merely the case that  $\tilde{\mathbb{E}}[\text{clique size}] \geq \omega$ , and it is far from clear how to upgrade the SOS solution so as to actually satisfy the constraint “clique size =  $\omega$ ” with pseudovariance zero. Besides being an improvement for its own sake, it would be very desirable to have such an SOS solution for the purposes of further reductions; for example, it would greatly simplify the recent proofs of SOS lower bounds for approximate Nash welfare in [15]. It also seems it might be useful for tackling SOS lower bounds for coloring and stochastic block models.



Finally, we leave as open one more “hard constraint” challenge that arises even in the simple context of random 3XOR or 3SAT. Suppose one tried to refute random  $m$ -constraint 3XOR instances by trying to refute the following statement for all quadruples  $(k_{001}, k_{010}, k_{100}, k_{111})$  that sum to  $m$ :

“exactly  $k_a$  constraints are satisfied with assignment  $a$ ”, for each  $a \in \{001, 010, 100, 111\}$ .

As far as we know, constant-degree SOS may succeed with this strategy when  $m = O(n)$ . It is natural to believe that there is (whp) an  $\Omega(n)$ -degree SOS pseudodistribution that satisfies all of the above constraints with pseudovariance zero when  $k_{001} = k_{010} = k_{100} = k_{111} = m/4$ , but we do not know how to construct one.

---

## References

- 1 Per Austrin and Johan Håstad. Randomly supported independence and resistance. *SIAM Journal on Computing*, 40(1):1–27, 2011.
- 2 Boaz Barak, Fernando Brandão, Aram Harrow, Jonathan Kelner, David Steurer, and Yuan Zhou. Hypercontractivity, Sum-of-Squares Proofs, and their Applications. In *Proc. of the 44th Annual ACM Symposium on Theory of Computing*, pages 307–326, 2012.
- 3 Boaz Barak, Siu On Chan, and Pravesh Kothari. Sum of Squares Lower Bounds from Pairwise Independence. In *Proc. of the 47th Annual ACM Symposium on Theory of Computing*, pages 97–106, 2015.
- 4 Boaz Barak, Samuel Hopkins, Jonathan Kelner, Pravesh Kothari, Ankur Moitra, and Aaron Potechin. A Nearly Tight Sum-of-Squares Lower Bound for the Planted Clique Problem. In *Proc. of the 57th Annual IEEE Symposium on Foundations of Computer Science*, 2016.
- 5 Boaz Barak and David Steurer. Proofs, beliefs, and algorithms through the lens of sum-of-squares, 2016. URL: <http://sumofsquares.org/public/index.html>.
- 6 Irit Dinur, Ehud Friedgut, Guy Kindler, and Ryan O’Donnell. On the Fourier tails of bounded functions over the discrete cube. *Israel Journal of Mathematics*, 160(1):389–412, 2007.
- 7 Uriel Feige. Relations between average case complexity and approximation complexity. In *Proc. of the 34th Annual ACM Symposium on Theory of Computing*, pages 543–543, 2002.
- 8 Dima Grigoriev. Complexity of Positivstellensatz proofs for the knapsack. *Computational Complexity*, 10(2):139–154, 2001.
- 9 Dima Grigoriev. Linear lower bound on degrees of Positivstellensatz calculus proofs for the parity. *Theoretical Computer Science*, 259(1-2):613–622, 2001.
- 10 Dima Grigoriev, Edward Hirsch, and Dmitrii Pasechnik. Complexity of semialgebraic proofs. *Moscow Mathematical Journal*, 2(4):647–679, 2002.
- 11 Venkatesan Guruswami, Ali Kemal Sinop, and Yuan Zhou. Constant factor Lasserre integrality gaps for graph partitioning problems. *SIAM Journal on Optimization*, 24(4):1698–1717, 2014.
- 12 Johan Håstad. Some optimal inapproximability results. *Journal of the ACM*, 48(4):798–859, 2001.
- 13 Subhash Khot. Ruling out PTAS for Graph Min-Bisection, Dense  $k$ -Subgraph, and Bipartite Clique. *SIAM Journal on Computing*, 36(4):1025–1071, 2006.
- 14 Pravesh Kothari, Ryuhei Mori, Ryan O’Donnell, and David Witmer. Sum of squares lower bounds for refuting any CSP. In *Proc. of the 49th Annual ACM Symposium on Theory of Computing*, pages 132–145, 2017.
- 15 Pravesh K. Kothari and Ruta Mehta. Sum-of-squares meets Nash: lower bounds for finding any equilibrium. In *Proc. of the 50th Annual ACM Symposium on Theory of Computing*, pages 1241–1248, 2018.

- 16 Ryan O’Donnell. SOS is not obviously automatizable, even approximately. In *Proc. of the 8th Annual Innovations in Theoretical Computer Science conference*, 2017.
- 17 Ryan O’Donnell and John Wright. A new point of NP-hardness for Unique-Games. In *Proc. of the 44th Annual ACM Symposium on Theory of Computing*, pages 289–306, 2012.
- 18 Prasad Raghavendra and Benjamin Weitz. On the Bit Complexity of Sum-of-Squares Proofs. Technical report, arXiv, 2017. [arXiv:1702.05139](https://arxiv.org/abs/1702.05139).
- 19 Grant Schoenebeck. Linear Level Lasserre Lower Bounds for Certain  $k$ -CSPs. In *Proc. of the 49th Annual IEEE Symposium on Foundations of Computer Science*, pages 593–602, 2008.
- 20 Luca Trevisan, Gregory Sorkin, Madhu Sudan, and David Williamson. Gadgets, Approximation, and Linear Programming. *SIAM Journal on Computing*, 29(6):2074–2097, 2000.
- 21 Madhur Tulsiani. CSP gaps and reductions in the Lasserre hierarchy. In *Proc. of the 41st Annual ACM Symposium on Theory of Computing*, pages 303–312, 2009.



# Semi-Online Bipartite Matching

**Ravi Kumar**

Google, Mountain View, CA, USA  
ravi.k53@gmail.com

**Manish Purohit**

Google, Mountain View, CA, USA  
mpurohit@google.com

**Aaron Schild**

University of California, Berkeley, CA, USA  
aschild@berkeley.edu

**Zoya Svitkina**

Google, Mountain View, CA, USA  
zoya@cs.cornell.edu

**Erik Vee**

Google, Mountain View, CA, USA  
erikvee@google.com

---

## Abstract

In this paper we introduce the *semi-online* model that generalizes the classical online computational model. The semi-online model postulates that the unknown future has a predictable part and an adversarial part; these parts can be arbitrarily interleaved. An algorithm in this model operates as in the standard online model, i.e., makes an irrevocable decision at each step.

We consider bipartite matching in the semi-online model. Our main contributions are competitive algorithms for this problem and a near-matching hardness bound. The competitive ratio of the algorithms nicely interpolates between the truly offline setting (i.e., no adversarial part) and the truly online setting (i.e., no predictable part).

**2012 ACM Subject Classification** Theory of computation → Online algorithms

**Keywords and phrases** Semi-Online Algorithms, Bipartite Matching

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2019.50

**Related Version** A full version of the paper is available at <https://arxiv.org/abs/1812.00134>.

**Acknowledgements** We thank Michael Kapralov for introducing us to the bipartite matching skeleton decomposition in [7].

## 1 Introduction

Modeling future uncertainty in data while ensuring that the model remains both realistic and tractable has been a formidable challenge facing the algorithms research community. One of the more popular, and reasonably realistic, such models is the online computational model. In its classical formulation, data arrives one at a time and upon each arrival, the algorithm has to make an irrevocable decision agnostic of future arrivals. Online algorithms boast a rich literature and problems such as caching, scheduling and matching, each of which



© Ravi Kumar, Manish Purohit, Aaron Schild, Zoya Svitkina, and Erik Vee;  
licensed under Creative Commons License CC-BY

10th Innovations in Theoretical Computer Science (ITCS 2019).

Editor: Avrim Blum; Article No. 50; pp. 50:1–50:20



Leibniz International Proceedings in Informatics

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

abstracts common practical scenarios, have been extensively investigated [4, 20]. Competitive analysis, which measures how well an online algorithm performs compared to the best offline algorithm that knows the future, has been a linchpin in the study of online algorithms.

While online algorithms capture some aspect of the future uncertainty in the data, the notion of competitive ratio is inherently worst-case and hence the quantitative guarantees it offers are often needlessly pessimistic. A natural question that then arises is : how can we avoid modeling the worst-case scenario in online algorithms? Is there a principled way to incorporate some knowledge we have about the future? There have been a few efforts trying to address this point from different angles. One line of attack has been to consider oracles that offer some advice on the future; such oracles, for instance, could be based on machine-learning methods. This model has been recently used to improve the performance of online algorithms for reserve price optimization, caching, ski-rental, and scheduling [19, 16, 14]. Another line of attack posits a distribution on the data [5, 17, 21] or the arrival model; for instance, random arrival models have been popular in online bipartite matching and are known to beat the worst-case bounds [10, 18]. A different approach is to assume a distribution on future inputs; the field of stochastic online optimization focuses on this setting [8]. The advice complexity model, where the partial information about the future is quantified as advice bits to an online algorithm, has been studied as well in complexity theory [3].

In this work we take a different route. At a very high level, the idea is to tease the future data apart into a predictable subset and the remaining adversarial subset. As the names connote, the algorithm can be assumed to know everything about the former but nothing about the latter. Furthermore, the predictable and adversarial subsets can arrive arbitrarily interleaved yet the algorithm still has to operate as in the classical online model, i.e., make irrevocable decisions upon each arrival. Our model thus offers a natural interpolation between the traditional offline and online models; we call ours the *semi-online* model. Our goal is to study algorithms in the semi-online model and to analyze their competitive ratios; unsurprisingly, the bounds depend on the size of the adversarial subset. Ideally, the competitive ratio should approach the offline optimum bounds if the adversarial fraction is vanishing and should approach the online optimum bounds if the predictable fraction is vanishing.

**Bipartite matching.** As a concrete problem in the semi-online setting, we focus on bipartite matching. In the well-known online version of the problem, which is motivated by online advertising, there is a bipartite graph with an offline side that is known in advance and an online side that is revealed one node at a time together with its incident edges. In the semi-online model, the nodes in the online side are partitioned into a predicted set of size  $n - d$  and an adversarial set of size  $d$ . The algorithm knows the incident edges of all the nodes in the former but nothing about the nodes in the latter. We can thus also interpret the setting as online matching with partial information and predictable uncertainty (pardon the oxymoron). In online advertising applications, there are many predictably unpredictable events. For example, during the soccer world cup games, we know the nature of web traffic will be unpredictable but nothing more, since the actual characteristics will depend on how the game progresses and which team wins.

We also consider a variant of semi-online matching in which the algorithm does not know which nodes are predictable and which are adversarial. In other words, the algorithm receives a prediction for all online nodes, but the predictions are correct only for some  $n - d$  of them. We call this the *agnostic* case.

**Main results.** In this paper, we assume that the optimum solution on the bipartite graph, formed by the offline nodes on one side and by the predicted and adversarial nodes on the other, is a perfect matching<sup>1</sup>. We present two algorithms and a hardness result for the semi-online bipartite matching problem. Let  $\delta = d/n$  be the fraction of adversarial nodes. The Iterative Sampling algorithm, described in Section 3, obtains a competitive ratio of  $(1 - \delta + \frac{\delta^2}{2}(1 - 1/e))^2$ . This algorithm “reserves” a set of offline nodes to be matched to the adversarial nodes by repeatedly selecting a random offline node that is unnecessary for matching the predictable nodes. It is easy to see that algorithms that deterministically reserve a set of offline nodes can easily be thwarted by the adversary.

The second algorithm, described in Section 4, achieves an improved competitive ratio of  $(1 - \delta + \delta^2(1 - 1/e))$ . This algorithm samples a maximum matching in the predicted graph by first finding a matching skeleton [7, 15] and then sampling a matching from each component in the skeleton using the dependent rounding scheme of [6]. This allows us to sample a set of offline nodes that, in expectation, has a large overlap with the set matched to adversarial nodes in the optimal solution. Surprisingly, in Section 5, we show that it is possible to sample from arbitrary set systems so that the same “large overlap” property is maintained. We prove the existence of such distributions using LP duality and believe that this result may be of independent interest.

To complement the algorithms, in Section 6 we obtain a hardness result showing that the above competitive ratios are near-optimal. In particular, no randomized algorithm can achieve a competitive ratio better than  $(1 - \delta e^{-\delta}) \approx (1 - \delta + \delta^2 - \delta^3/2 + \dots)$ . Note that this expression coincides with the best offline bound (i.e., 1) and the best online bound (i.e.,  $1 - 1/e$ ) at the extremes of  $\delta = 0$  and  $\delta = 1$ , respectively. We conjecture this to be the optimal bound for the problem.

**Extensions.** In Section 7, we explore variants of the semi-online matching model, including the agnostic version and fractional matchings, and present upper and lower bounds in those settings. To illustrate the generality of our semi-online model, we consider a semi-online version of the classical ski rental problem. In this version, the skier knows whether or not she’ll ski on certain days while other days are uncertain. Interestingly, there is an algorithm with a competitive ratio of the same form as our hardness result for matchings, namely  $1 - (1 - x)e^{-(1-x)}$ , where  $(1 - x)$  is a parameter analogous to  $\delta$  in the matching problem. We wonder if this form plays a role in semi-online algorithms similar to what  $(1 - 1/e)$  has in many online algorithms [11].

**Other related work.** The use of (machine learned) predictions for revenue optimization problems was first proposed in [19]. The concepts were formalized further and applied to online caching in [16] and ski rental and non-clairvoyant scheduling in [14]. Online matching with forecasts was first studied in [22]; however, that paper is on forecasting the demands rather than the structure of the graph as in our case. The problem of online matching when edges arrive in batches was considered in [15] where a  $(1/2 + 2^{-O(s)})$ -competitive ratio is shown, with  $s$  the number of stages. However, the batch framework differs from ours in that in our case, the nodes arrive one at a time and are arbitrarily interleaved.

<sup>1</sup> Our techniques extend to the case without a perfect matching; we defer the proof of the general case to the full version of the paper.

<sup>2</sup> Observe that an algorithm that ignores all the adversarial nodes and outputs a maximum matching in the predicted graph achieves a competitive ratio of only  $1 - \delta$ .

There has been a lot of work on online bipartite matching and its variants. The RANKING algorithm [13] selects a random permutation of the offline nodes and then matches each online node to its unmatched neighbor that appears earliest in the permutation. It is well-known to obtain a competitive ratio of  $(1 - 1/e)$ , which is best possible. For a history of the problem and significant advances, see the monograph [20]. The ski rental problem has also been extensively studied; the optimal randomized algorithm has ratio  $e/(e - 1)$  [12]. The term “semi-online” has been used in scheduling when an online scheduler knows the sum of the jobs’ processing times (e.g., see [1]) and in online bin-packing when a lookahead of the next few elements is available (e.g., see [2]); our use of the term is more quantitative in nature.

## 2 Model

We now formally define the *semi-online bipartite matching* problem. We have a bipartite graph  $G = (U, V, E_G)$  where  $U$  is the set of nodes available offline and nodes in  $V$  arrive online. Further, the online set  $V$  is partitioned into the *predicted* nodes  $V_P$  and the *adversarial* nodes  $V_A$ . The *predicted graph*  $H = (U, V_P, E_H)$  is the subgraph of  $G$  induced on the nodes in  $U$  and  $V_P$ . Initially, an algorithm only knows  $H$  and is unaware of edges between  $U$  and  $V_A$ . The algorithm is allowed to preprocess  $H$  before any online node actually arrives. In the online phase, at each step, one node of  $V$  is revealed with its incident edges, and has to be either irrevocably matched to some node in  $U$  or abandoned. Nodes of  $V$  are revealed in an arbitrary order<sup>3</sup> and the process continues until all of  $V$  has been revealed.

We note that when a node  $v \in V$  is revealed, the algorithm can “recognize” it by its edges, i.e., if there is some node  $v' \in V_P$  that has the same set of neighbors as  $v$  and has not been seen before, then  $v$  can be assumed to be  $v'$ . There could be multiple identical nodes in  $V_P$ , but it is not important to distinguish between them. If an online node comes that is not in  $V_P$ , then the algorithm can safely assume that it is from  $V_A$ . (In Section 7, we consider a model where the predicted graph can have errors and hence this assumption is invalid.)

We introduce a quantity  $\delta$  to measure the level of knowledge that the algorithm has about the input graph  $G$ . Competitive ratios that we obtain are functions of  $\delta$ . For any graph  $I$ , let  $\nu(I)$  denote the size of the maximum matching in  $I$ . Then we define  $\delta = \delta(G) = 1 - \frac{\nu(H)}{\nu(G)}$ . Intuitively, the closer  $\delta$  is to 0, the more information the predicted graph  $H$  contains and the closer the instance is to an offline problem. Conversely,  $\delta$  close to 1 indicates an instance close to the pure online setting. Note that the algorithm does not necessarily know  $\delta$ , but we use it in the analysis to bound the competitive ratio. For convenience, in this paper we assume that the input graph  $G$  contains a perfect matching. Let  $n = |U| = |V|$  be the number of nodes on each side and  $d = |V_A|$  be the number of adversarial online nodes. In this case,  $\delta$  simplifies to be the fraction of online nodes that are adversarial, i.e.,  $\delta = \frac{|V_A|}{|V|} = \frac{d}{n}$ .

## 3 Iterative Sampling Algorithm

In this section we give a simple polynomial time algorithm for bipartite matching in the semi-online model. We describe the algorithm in two phases: a *preprocessing phase* that finds a maximum matching  $M$  in the predicted graph  $H$  and an *online phase* that processes each node upon its arrival to find a matching in  $G$  that extends  $M$ .

<sup>3</sup> The arrival order can be adversarial, including interleaving the nodes in  $V_P$  and  $V_A$ .



**Algorithm 1:** Iterative Sampling: Preprocessing Phase.

---

```

Function: Preprocess( $H$ ):
  Data: Predicted graph  $H$ 
  Result: Maximum matching in  $H$ , a sequence of nodes from  $U$ 

  Let  $H_0 \leftarrow H, U_0 \leftarrow U$ ;
  for  $i = 1, 2, \dots, d$  do
     $U_i \leftarrow \{u \in U_{i-1} \mid \nu(H_{i-1} \setminus \{u\}) = n - d\}$ ;          /* Set of nodes whose
    removal does not change the size of the maximum matching. */
    Let  $u_i$  be a uniformly random node in  $U_i$ ;
     $H_i \leftarrow H_{i-1} \setminus \{u_i\}$ ;
   $M \leftarrow$  Arbitrary maximum matching in  $H_d$ ;
   $R \leftarrow$  Uniformly random permutation of  $\{u_1, \dots, u_d\}$ ;
  return  $M, R$ 

```

---

**Algorithm 2:** Online Phase.

---

```

 $M, R \leftarrow$  Preprocess( $H$ );
for  $v \in V$  arriving online do
  if  $v \in V_P$  then                                          /* predicted node */
    Match  $v$  to  $M(v)$ ;
  else                                                         /* adversarial node */
    Match  $v$  to the first unmatched neighbor in  $R$ , if one exists; /* RANKING */

```

---

**Preprocessing Phase**

The goal of the preprocessing phase is to find a maximum matching in the predicted graph  $H$ . However, if we deterministically choose a matching, the adversary can set up the neighborhoods of  $V_A$  so that all the neighbors of  $V_A$  are used in the chosen matching, and hence the algorithm is unable to match any node from  $V_A$ . Algorithm 1 describes our algorithm to sample a (non-uniform) random maximum matching from  $H$ .

**Online Phase**

In the online phase nodes from  $V$  arrive one at a time and we are required to maintain a matching such that the online nodes are either matched irrevocably or dropped. In this phase, we match the nodes in  $V_P$  as per the matching  $M$  obtained from the preprocessing phase, i.e., we match  $v \in V_P$  to node  $M(v) \in U$  where  $M(v)$  denotes the node matched to  $v$  by matching  $M$ . The adversarial nodes in  $V_A$  are matched to nodes in  $R$  that are not used by  $M$  using the RANKING algorithm [13]. Algorithm 2 describes the complete online phase of our algorithm.

**Analysis**

For the sake of analysis, we construct a sequence of matchings  $\{M_i^*\}_{i=0}^d$  as follows. Let  $M_0^*$  be an arbitrary perfect matching in  $G$ . For  $i \geq 1$ , by definition of  $U_i$ , there exists a matching  $M_i'$  in  $H_i$  of size  $n - d$  that does not match node  $u_i$ . Hence,  $M_i' \cup M_{i-1}^*$  is a union of disjoint paths and cycles such that  $u_i$  is an endpoint of a path  $P_i$ . Let  $M_i^* = M_{i-1}^* \oplus P_i$ ,

i.e. obtain  $M_i^*$  from  $M_{i-1}^*$  by adding and removing alternate edges from  $P_i$ . It's easy to verify that  $M_i^*$  is indeed a matching and  $|M_i^*| \geq |M_{i-1}^*| - 1$ . Since  $|M_0^*| = n$ , this yields  $|M_i^*| \geq n - i$ ,  $\forall 0 \leq i \leq d$ . Further, by construction,  $M_i^*$  does not match any nodes in  $\{u_1, \dots, u_i\}$ .

► **Lemma 1.** *For all  $0 \leq i \leq d$ , all nodes  $v \in V_P$  are matched by  $M_i^*$ . Further,  $|M_i^*(V_A)| \geq d - i$ , i.e. at least  $d - i$  adversarial nodes are matched by  $M_i^*$ .*

**Proof.** We prove the claim by induction. Since  $M_0^*$  is a perfect matching, the base case is trivially true. By the induction hypothesis, we assume that  $M_{i-1}^*$  matches all of  $V_P$ . Recall that  $M_i'$  also matches all of  $V_P$  and  $M_i^* = M_{i-1}^* \oplus P_i$  where  $P_i$  is a maximal path in  $M_i' \cup M_{i-1}^*$ . Since each node  $v \in V_P$  has degree 2 in  $M_i' \cup M_{i-1}^*$ ,  $v$  cannot be an end point of  $P_i$ . Hence, all nodes  $v \in V_P$  remain matched in  $M_i^*$ . Further, we have  $|M_i^*(V_A)| = |M_i^*| - |M_i^*(V_P)| \geq (n - i) - (n - d) = d - i$  as desired. ◀

Equipped with the sequence of matchings  $M_i^*$ , we are now ready to prove that, in expectation, a large matching exists between the set  $R$  of nodes left unmatched by the preprocessing phase and the set  $V_A$  of adversarial nodes.

► **Lemma 2.**  $\mathbb{E}[\nu(G[R \cup V_A])] \geq d^2/(2n)$  where  $G[R \cup V_A]$  is the graph induced by the reserved vertices  $R$  and the adversarial vertices  $V_A$ .

**Proof.** We construct a sequence of sets of edges  $\{N_i\}_{i=0}^d$  as follows. Let  $N_0 = \emptyset$ . If  $M_{i-1}^*(u_i) \in V_A$ , let  $e_i = \{u_i, M_{i-1}^*(u_i)\}$  be the edge of  $M_{i-1}^*$  incident with  $u_i$  and let  $N_i = N_{i-1} \cup \{e_i\}$ . Otherwise, let  $N_i = N_{i-1}$ . In other words, if the node  $u_i$  chosen during the  $i^{\text{th}}$  step is matched to an adversarial node by the matching  $M_{i-1}^*$ , add the matched edge to set  $N_i$ .

We show by induction that  $N_i$  is a matching for all  $i \geq 0$ .  $N_0$  is clearly a matching. When  $i > 0$ , either  $N_i = N_{i-1}$  (in which case we are done by the inductive hypothesis), or  $N_i = N_{i-1} \cup \{e_i\}$ . Let  $e_i = (u_i, v_i)$  and consider any other edge  $e_j = (u_j, v_j) \in N_{i-1}$ . Since  $u_j \notin H_{i-1}$ , we have  $u_j \neq u_i$ . By definition, node  $v_i$  is matched in  $M_{i-1}^*$ . By construction, this implies that  $v_i$  must be matched in all previous matchings in this sequence, in particular,  $v_i$  must be matched in  $M_j^*$  (since a node  $v \in V_A$  that is unmatched in  $M_{k-1}^*$  can never be matched by  $M_k^*$ ). However, since  $v_j = M_{j-1}^*(u_j)$ , the matching  $M_j^* = M_{j-1}^* \setminus \{e_j\}$  and hence  $v_j$  is not matched in  $M_j^*$ . Hence  $v_i \neq v_j$ . Thus we have shown that  $e_i$  does not share an endpoint with any  $e_j \in N_{i-1}$  and hence  $N_i$  is a matching.

By linearity of expectation we have the following.

$$\mathbb{E}[|N_i|] = \mathbb{E}[|N_{i-1}|] + \Pr_{u_i}[M_{i-1}^*(u_i) \in V_A]$$

However, by Lemma 1, since  $M_{i-1}^*$  matches all of  $V_P$ , we must have  $M_{i-1}^*(V_A) \subseteq U_i$ . Hence,

$$\mathbb{E}[|N_i|] \geq \mathbb{E}[|N_{i-1}|] + \frac{|M_{i-1}^*(V_A)|}{|U_i|} \geq \mathbb{E}[|N_{i-1}|] + \frac{d - (i - 1)}{n}$$

Solving the recurrence with  $|N_0| = 0$  gives

$$\mathbb{E}[|N_d|] \geq \sum_{i=1}^d \frac{i}{n} \geq \frac{d(d+1)}{2n}$$

The lemma follows since  $N_d$  is a matching in  $G[R \cup V_A]$ . ◀

► **Theorem 3.** *There is a randomized algorithm for the semi-online bipartite matching problem with a competitive ratio of at least  $(1 - \delta + (\delta^2/2)(1 - 1/e))$  in expectation.*

**Proof.** Algorithm 1 guarantees that the matching  $M$  found in the preprocessing phase matches all predicted nodes and has size  $n - d = n(1 - \delta)$ . Further, in the online phase, we use the RANKING [13] algorithm on the graph  $G[R \cup V_A]$ . Since RANKING is  $(1 - 1/e)$ -competitive, the expected number of adversarial nodes matched is at least  $(1 - 1/e)\nu(G[R \cup V_A])$ . By Lemma 2, this is at least  $(1 - 1/e)(\frac{d^2}{2n}) = (\delta^2 n/2)(1 - 1/e)$ .

Therefore, the total matching has expected size  $n(1 - \delta + (\delta^2/2)(1 - 1/e))$  as desired. ◀

Using a more sophisticated analysis, we can show that the iterative sampling algorithm yields a tighter bound of  $(1 - \delta + \delta^2/2 - \delta^3/2)$ . However we omit the proof because the next section presents an algorithm with an even better guarantee.

## 4 Structured Sampling

In this section we give a polynomial time algorithm for the semi-online bipartite matching that yields an improved competitive ratio of  $(1 - \delta + \delta^2(1 - 1/e))$ . We first discuss the main ideas in Section 4.1 and then describe the algorithm and its analysis in Section 4.2.

► **Theorem 4.** *There is a randomized algorithm for the semi-online bipartite matching problem with a competitive ratio of at least  $(1 - \delta + \delta^2(1 - 1/e))$  in expectation.*

### 4.1 Main Ideas and Intuition

As with the iterative sampling algorithm, we randomly choose a matching of size  $n - d$  (according to some distribution), and define the *reserved* set  $R$  to be the set of offline nodes that are not matched. As online nodes arrive, we follow the matching for the predicted nodes; for adversarial nodes, we run the RANKING algorithm on the reserved set  $R$ .

Let  $M^*$  be a perfect matching in  $G$ . For a set of nodes  $S$ , let  $M^*(S)$  denote the set of nodes matched to them by  $M^*$ . Call a node  $u \in U$  *marked* if it is in  $M^*(V_A)$ , i.e., it is matched to an adversarial node by the optimal solution. We argue that the number of marked nodes in the set  $R$  chosen by our algorithm is at least  $d^2/n$  in expectation. Since RANKING finds a matching of at least a factor  $(1 - 1/e)$  of optimum in expectation, this means that we find a matching of size at least  $d^2/n \cdot (1 - 1/e)$  on the reserved nodes in expectation. Combining this with the matching of size  $n - d$  on the predicted nodes, this gives a total of  $n - d + d^2/n \cdot (1 - 1/e) = n(1 - \delta + \delta^2(1 - 1/e))$ .

The crux of the proof lies in showing that  $R$  contains many marked nodes. Ideally, we would like to choose a random matching of size  $n - d$  in such a way that each node of  $U$  has probability  $d/n$  of being in  $R$ . Since there are  $d$  marked nodes total,  $R$  would contain  $d^2/n$  of them in expectation. However, such a distribution over matchings does not always exist.

Instead, we use a graph decomposition to guide the sampling process. The marginal probabilities for nodes of  $U$  to be in  $R$  may differ, but nevertheless  $R$  gets the correct total number of marked nodes in expectation.  $H$  is decomposed into bipartite pairs  $(S_i, T_i)$ , with  $|S_i| \leq |T_i|$ , so that the sets  $S_i$  partition  $V_P$  and the sets  $T_i$  partition  $U$ . This decomposition allows one to choose a random matching between  $S_i$  and  $T_i$  of size  $|S_i|$  so that each node in  $T_i$  is reserved with the same probability. Letting  $n_i = |T_i|$  and  $d_i = |T_i| - |S_i|$ , this probability is precisely  $d_i/n_i$ . Finally, we argue that the adversary can do no better than to mark  $d_i$  nodes in  $T_i$ , for each  $i$ . Hence, the expected number of nodes in  $R$  that are marked is at least  $\sum_i (d_i^2/n_i)$ , which we lower bound by  $d^2/n$ .

## 4.2 Proof of Theorem 4

We decompose the graph  $H$  into more structured pieces using a construction from [7] and utilize the key observation that the decomposition implies a *fractional matching*. Recall that a fractional matching is a function  $f$  that assigns a value in  $[0, 1]$  to each edge in a graph, with the property that  $\sum_{e \ni v} f(e) \leq 1$  for all nodes  $v$ . The quantity  $\sum_{e \ni v} f(e)$  is referred to as the *fractional degree* of  $v$ . We use  $\Gamma(S)$  to denote the set of neighbors of nodes in  $S$ .

► **Lemma 5** (Restatement of Lemma 2 from [15]). *Given a bipartite graph  $H = (U, V_P, E_H)$  with  $|U| \geq |V_P|$  and a maximum matching of size  $|V_P|$ , there exists a partition of  $V_P$  into sets  $S_0, \dots, S_m$  and a partition of  $U$  into sets  $T_0, \dots, T_m$  for some  $m$  such that the following holds:*

- $\Gamma(\bigcup_{i < j} S_i) = \bigcup_{i < j} T_i$  for all  $j$ .
- For all  $i < j$ ,  $\frac{|S_i|}{|T_i|} > \frac{|S_j|}{|T_j|}$ .
- There is a fractional matching in  $H$  of size  $|V_P|$ , where for all  $i$ , the fractional degree of each node in  $S_i$  is 1 and the fractional degree of each node in  $T_i$  is  $|S_i|/|T_i|$ . In this matching, nodes in  $S_i$  are only matched with nodes in  $T_i$  and vice versa.

Further, the  $(S_i, T_i)$  pairs can be found in polynomial time.

In [7] and [15], the sets in the decomposition with  $|S_i| < |T_i|$  are indexed with positive integers  $i > 0$ , the sets with  $|S_0| = |T_0|$  get an index of 0, and the ones with  $|S_i| > |T_i|$  get negative indices  $i < 0$ . Under our assumption that  $H$  supports a matching that matches all nodes of  $V_P$ , the decomposition does not contain sets with  $|S_i| > |T_i|$ , as the first such set would have  $|S_i| > |\Gamma(S_i)|$ , violating Hall's theorem. So we start the indices from 0.

Equipped with this decomposition, we choose a random matching between  $S_i$  and  $T_i$  such that each node in  $T_i$  is reserved<sup>4</sup> with the same probability. Since each  $(S_i, T_i)$  pair has a fractional matching, the dependent randomized rounding scheme of [6] allows us to do exactly that.

► **Lemma 6.** *Fix an index  $i$  and let  $S_i, T_i$  be defined as in Lemma 5. Then there is a distribution over matchings with size  $|S_i|$  between  $S_i$  and  $T_i$  such that for all  $u \in T_i$ , the probability that the matching contains  $u$  is  $|S_i|/|T_i|$ .*

**Proof.** Given any bipartite graph  $G'$  and a fractional matching over  $G'$ , the dependent rounding scheme of Gandhi et. al. [6] yields an integral matching such that the probability that any node  $v \in G'$  is matched exactly equals its fractional degree. Since Lemma 5 guarantees a fractional matching such that the fractional degree of each node in  $S_i$  is 1 and the fractional degree of each node in  $T_i$  is  $|S_i|/|T_i|$ , the lemma follows. ◀

We are now ready to complete the description of our algorithm. Algorithm 3 is the preprocessing phase, while the online phase remains the same as earlier (Algorithm 2). In the preprocessing phase, we find a decomposition of the predicted graph  $H$ , and sample a matching using Lemma 6 for each component in the decomposition. In the online phase, we match all predicted online nodes using the sampled matching and use RANKING to match the adversarial online nodes.

Let  $n_i = |T_i|$  and  $d_i = |T_i| - |S_i|$ , and let  $R_i = R \cap T_i$  be the set of reserved nodes in  $T_i$ . Then Lemma 6 says that each node in  $T_i$  lands in  $R_i$  with probability  $d_i/n_i$  (although not independently). We now argue in Lemmas 7 and 8 that the adversary can do no better than to choose  $d_i$  marked nodes in each  $T_i$ .

<sup>4</sup> Recall that we say a node  $u$  is reserved by an algorithm if  $u$  is *not* matched in the predicted graph  $H$ .

---

**Algorithm 3:** Structured Sampling: Preprocessing Phase.
 

---

**Function:** Preprocess( $H$ ):**Data:** Predicted graph  $H$ **Result:** Maximum matching in  $H$ , sequence of nodes from  $U$ Decompose  $H$  into  $\{(S_i, T_i)\}_{i=0}^m$  pairs using Lemma 5. $M_i \leftarrow$  Random matching between  $S_i$  and  $T_i$  using Lemma 6 $M \leftarrow \bigcup_i M_i$ Let  $R_{\text{set}} \subseteq U$  be the set of nodes unmatched by  $M$  $R \leftarrow$  Uniformly random permutation of  $R_{\text{set}}$ **return**  $M, R$ 

► **Lemma 7.** Let  $\ell_i = |M^*(V_A) \cap T_i|$ . That is, let  $\ell_i$  be the number of marked nodes in  $T_i$ . Then for all  $t \geq 0$ ,

$$\sum_{i \leq t} \ell_i \leq \sum_{i \leq t} d_i$$

**Proof.** Fix  $t \leq m$ , and let  $U' = U - \bigcup_{i \leq t} T_i$ . Since there is a perfect matching in the realized graph  $G$ , Hall's Theorem guarantees that there must be at least  $|U'| - |\Gamma_H(U')|$  marked nodes in  $U'$  where  $\Gamma_H(U')$  denotes the set of neighbors of  $U'$  in the predicted graph  $H$ . That is,

$$\sum_{i > t} \ell_i \geq |U'| - |\Gamma_H(U')|$$

But Lemma 5 tells us that  $\Gamma(\bigcup_{i \leq t} S_i) = \bigcup_{i \leq t} T_i$ , hence there is no edge between  $U'$  and  $\bigcup_{i \leq t} S_i$ . That is,  $\Gamma_H(U') \subseteq V_P - \bigcup_{i \leq t} S_i$ . Hence,

$$|\Gamma_H(U')| \leq |V_P| - \sum_{i \leq t} |S_i| = n - d - \sum_{i \leq t} (n_i - d_i)$$

Further,  $|U'| = |U| - |\bigcup_{i \leq t} T_i| = n - \sum_{i \leq t} n_i$ . Putting this together,

$$\begin{aligned} \sum_{i > t} \ell_i &\geq |U'| - |\Gamma_H(U')| \\ &\geq n - \sum_{i \leq t} n_i - \left( n - d - \sum_{i \leq t} (n_i - d_i) \right) \\ &= d - \sum_{i \leq t} d_i \end{aligned}$$

Recalling that  $\sum_{i \leq m} \ell_i = d$ , we see that  $\sum_{i \leq t} \ell_i = d - \sum_{i > t} \ell_i \leq \sum_{i \leq t} d_i$ , as desired. ◀

► **Lemma 8.** Let  $0 < a_0 \leq a_1 \leq \dots \leq a_m$  be a non-decreasing sequence of positive numbers, and  $\ell_0, \dots, \ell_m$  and  $k_0, \dots, k_m$  be non-negative integers, such that  $\sum_{i=0}^m \ell_i = \sum_{i=0}^m k_i$  and for all  $t \leq m$ ,  $\sum_{i \leq t} \ell_i \leq \sum_{i \leq t} k_i$ . Then

$$\sum_{i=0}^m \ell_i a_i \geq \sum_{i=0}^m k_i a_i.$$

**Proof.** We claim that for any fixed sequence  $k_0, \dots, k_m$ , the minimum of the left-hand side ( $\sum_i \ell_i a_i$ ) is attained when  $\ell_i = k_i$  for all  $i$ . Suppose for contradiction that  $\{\ell_i\}$  is the lexicographically-largest minimum-attaining assignment that is not equal to  $\{k_i\}$  and let  $j$  be the smallest index with  $\ell_j \neq k_j$ . It must be that  $\ell_j < k_j$  to satisfy  $\sum_{i \leq j} \ell_i \leq \sum_{i \leq j} k_i$ . Also,  $\sum_{i=0}^m \ell_i = \sum_{i=0}^m k_i$  implies that  $j < m$  and that there must be an index  $j' > j$  such that  $\ell_{j'} > k_{j'}$ . Let  $j'$  be the lowest such index.

Let  $\ell'_i = \ell_i$  for all  $i \notin \{j, j'\}$ . Set  $\ell'_j = \ell_j + 1$  and  $\ell'_{j'} = \ell_{j'} - 1$ . Notice that we still have  $\sum_{i \leq t} \ell'_i \leq \sum_{i \leq t} k_i$  for all  $t$  and  $\sum_{i=0}^m \ell'_i = \sum_{i=0}^m k_i$ , and  $\{\ell'_i\}$  is lexicographically larger than  $\{\ell_i\}$ . In addition,

$$\sum_i \ell'_i a_i = \sum_i \ell_i a_i + a_j - a_{j'} \leq \sum_i \ell_i a_i,$$

which is a contradiction. ◀

We need one last technical observation before the proof of the main result.

► **Lemma 9.** *Let  $d_i, n_i$  be positive numbers with  $\sum_i d_i = d$  and  $\sum_i n_i = n$ . Then*

$$\sum_i \frac{d_i^2}{n_i} \geq \frac{d^2}{n}$$

**Proof.** We invoke Cauchy-Schwartz, with vectors  $u$  and  $v$  defined by  $u_i = \frac{d_i}{\sqrt{n_i}}$  and  $v_i = \sqrt{n_i}$ . Since  $\|u\|^2 \geq |u \cdot v|^2 / \|v\|^2$ , the result follows. ◀

► **Theorem 10.** *Choose reserved set  $R$  according to Algorithm 2. Then the expected number of marked nodes in  $R$  is at least  $\delta^2 n$ . That is,  $|R \cap M^*(V_A)| \geq \delta^2 n$  in expectation.*

**Proof.** As in Lemma 7, let  $\ell_i = |M^*(V_A) \cap T_i|$ . Again, we have  $\sum_{i \leq t} \ell_i \leq \sum_{i \leq t} d_i$  for all  $t$  and  $\sum_{i \leq m} \ell_i = d = \sum_{i \leq m} d_i$ . For each  $i$ , the node  $u \in T_i$  is chosen to be in  $R$  with probability  $d_i/n_i$ , with the  $d_i/n_i$  forming an increasing sequence. So the expected size of  $|R \cap M^*(V_A)|$  is given by

$$\begin{aligned} \sum_i \frac{d_i}{n_i} \ell_i &\geq \sum_i \frac{d_i}{n_i} d_i && \text{by Lemma 8} \\ &\geq \frac{d^2}{n} && \text{by Lemma 9} \end{aligned}$$

Since  $\delta = d/n$ , the theorem follows. ◀

**Proof of Theorem 4.** The size of the matching, restricted to non-adversarial nodes, is  $\sum_i (n_i - d_i) = n - d = n - \delta n$ . Further, by Theorem 10, we have reserved at least  $\delta^2 n$  nodes that can be matched to the adversarial nodes. RANKING will match at least a  $(1 - 1/e)$  fraction of these in expectation. So in expectation, the total matching has size at least  $n - \delta n + \delta^2 n(1 - 1/e) = n(1 - \delta + \delta^2(1 - 1/e))$  as desired. ◀

## 5 Sampling From Arbitrary Set Systems

In Section 4, we used graph decomposition to sample a matching in the predicted graph such that, in expectation, there is a large overlap between the set of reserved (unmatched) nodes and the unknown set of marked nodes chosen by the adversary. In this section we prove the existence of probability distributions on sets, with this “large overlap” property, in settings more general than just bipartite graphs.

Let  $U$  be a universe of  $n$  elements and let  $\mathcal{S}$  denote a family of subsets of  $U$  with equal sizes, i.e.,  $|S| = d, \forall S \in \mathcal{S}$ . Suppose an adversary chooses a set  $T \in \mathcal{S}$ , which is unknown to us. Our goal is to find a probability distribution over  $\mathcal{S}$  such that the expected intersection size of  $T$  and a set sampled from this distribution is maximized. We prove in Theorem 11 that for any such set system, one can always guarantee that the expected intersection size is at least  $\frac{d^2}{n}$ .

The connection to matchings is as follows. Let  $U$ , the set of offline nodes in the matching problem, also be the universe of elements.  $\mathcal{S}$  is a collection of all maximal subsets  $R$  of  $U$  such that there is a perfect matching between  $U \setminus R$  and  $V_P$ . All these subsets have size  $d = |V_A| = \delta n$ . Notice that  $M^*(V_A)$  is one of the sets in  $\mathcal{S}$ , although of course we don't know which. What we would like is a distribution such that sampling a set  $R$  from it satisfies  $\mathbb{E}[|R \cap M^*(V_A)|] \geq d^2/n = \delta^2 n$ .

► **Theorem 11.** *For any set system  $(U, \mathcal{S})$  with  $|U| = n$  and  $|S| = d$  for all  $S \in \mathcal{S}$ , there exists a probability distribution  $\mathcal{D}$  over  $\mathcal{S}$  such that  $\forall T \in \mathcal{S}, \mathbb{E}_{S \sim \mathcal{D}}[|S \cap T|] \geq \frac{d^2}{n}$ .*

As an example, consider  $U = \{v, w, x, y, z\}$  and  $\mathcal{S} = \{\{v, w\}, \{w, x\}, \{x, y\}, \{y, z\}\}$ . Here  $n = 5$  and  $d = 2$ , so the theorem guarantees a probability distribution on the four sets such that each of them has an expected intersection size with the selected set of at least  $\frac{4}{5}$ . We can set  $\Pr[\{v, w\}] = \Pr[\{y, z\}] = \frac{3}{10}$  and  $\Pr[\{w, x\}] = \Pr[\{x, y\}] = \frac{1}{5}$ . Then the expected intersection size for the set  $\{v, w\}$  is  $\Pr[\{v, w\}] \cdot 2 + \Pr[\{w, x\}] \cdot 1 = \frac{4}{5}$  because the intersection size is 2 if  $\{v, w\}$  is picked and 1 if  $\{w, x\}$  is picked. Similarly, one can verify that the expected intersection for any set is at least  $\frac{4}{5}$ . However, in general, it is not trivial to find such a distribution via an explicit construction.

Theorem 11 is a generalization to Theorem 10, and we could have selected a matching and a reserved set  $R$  according to the methods used in its proof. Indeed, this gives the same competitive ratio. However, the set system generated by considering all matchings of size  $n - d$  is exponentially large in general. Hence the offline portion of the algorithm would not run in polynomial time.

## 5.1 Proof of Theorem 11

Let  $\mathcal{D}$  be a probability distribution over  $\mathcal{S}$  with the probability of choosing a set  $S$  denoted by  $p_S$ . Now, for any fixed set  $T \in \mathcal{S}$ , the expected intersection size is given by  $\mathbb{E}_{S \sim \mathcal{D}}[|S \cap T|] = \sum_{S \in \mathcal{S}} p_S \cdot |S \cap T| = \sum_{u \in T} \sum_{S \ni u} p_S$ . For a given set system  $(U, \mathcal{S})$ , consider the following linear program and its dual.

The primal constraints exactly capture the requirement that the expected intersection size is at least  $\frac{d^2}{n}$  for any choice of  $T$ . Thus, to prove the theorem, it suffices to show that the optimal primal solution has an objective value of at most 1. We show that any feasible solution to the dual linear program must have objective value at most 1 and hence the theorem follows from strong duality.

► **Lemma 12.** *For any set system  $(U, \mathcal{S})$ , the optimal solution to Dual-LP has objective value at most 1.*



$$\begin{array}{ll}
 \min \sum_{S \in \mathcal{S}} p_S & \max \sum_{T \in \mathcal{S}} q_T \\
 \text{s.t.} & \text{s.t.} \\
 \forall T \in \mathcal{S}, \sum_{u \in T} \sum_{S \ni u} p_S \geq \frac{d^2}{n} & \forall S \in \mathcal{S}, \sum_{u \in S} \sum_{T \ni u} q_T \leq \frac{d^2}{n} \quad (3) \\
 \forall S \in \mathcal{S}, p_S \geq 0 & \forall T \in \mathcal{S}, q_T \geq 0 \quad (4)
 \end{array}$$

■ Figure 1 Primal-LP.

■ Figure 2 Dual-LP.

**Proof.** Let  $\{q_T\}_{T \in \mathcal{S}}$  denote an optimal, feasible solution to Dual-LP. For any element  $u \in U$ , define  $w(u) = \sum_{T \ni u} q_T$  to be the total weight of all the sets that contain  $u$ . From the dual constraints, we have

$$\forall S \in \mathcal{S}, \sum_{u \in S} w(u) \leq \frac{d^2}{n}$$

Since each  $S \in \mathcal{S}$  has exactly  $d$  elements, we can rewrite the above as

$$\forall S \in \mathcal{S}, \sum_{u \in S} \left( w(u) - \frac{d}{n} \right) \leq 0$$

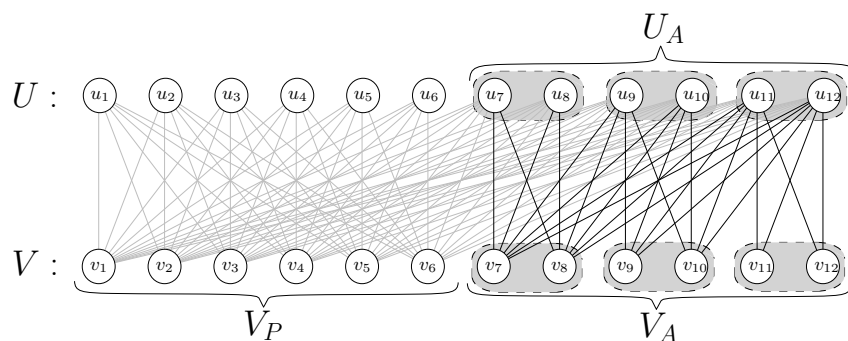
Multiplying each inequality by  $q_S$  and adding over all  $S \in \mathcal{S}$  yields

$$\begin{aligned}
 \sum_{S \in \mathcal{S}} \left( \sum_{u \in S} \left( w(u) - \frac{d}{n} \right) q_S \right) &\leq 0 \\
 \sum_{u \in U} \left( \sum_{S \ni u} \left( w(u) - \frac{d}{n} \right) q_S \right) &\leq 0 \\
 \sum_{u \in U} w(u) \left( w(u) - \frac{d}{n} \right) &\leq 0
 \end{aligned}$$

Using the fact that  $y(x - y) \leq x(x - y)$  for any two real numbers  $x$  and  $y$ , we get that  $\forall u \in U, \frac{d}{n}(w(u) - \frac{d}{n}) \leq w(u)(w(u) - \frac{d}{n})$ . Thus,

$$\begin{aligned}
 \sum_{u \in U} \frac{d}{n} \left( w(u) - \frac{d}{n} \right) &\leq 0 \\
 \sum_{u \in U} w(u) &\leq d \quad (5)
 \end{aligned}$$

On the other hand, we have  $\sum_{u \in U} w(u) = \sum_{u \in U} \sum_{T \ni u} q_T = \sum_{T \in \mathcal{S}} \sum_{u \in T} q_T = d \sum_{T \in \mathcal{S}} q_T$ . Inequality 5 then shows that  $\sum_{T \in \mathcal{S}} q_T \leq 1$ . ◀



■ **Figure 3** Hard instance for  $n = 12$ ,  $d = 6$ , and  $t = 1$ .

## 6 Hardness of Semi-Online Bipartite Matching

In this section, we show that no algorithm solving the semi-online bipartite matching problem can have a competitive ratio better than  $1 - \delta e^{-\delta}$ . The construction is similar in spirit to the original bound for online bipartite matching of [13]. However, rather than using a graph whose adjacency matrix is upper triangular, the core hardness comes from using a block upper triangular matrix.

### 6.1 Graph Construction

The constructed instance will have a perfect matching in  $G$ . Let  $d = |V_A| = \delta n$  be the number of adversarial online nodes and set  $t = d^{2/3}$  (it's only important that  $t = o(d)$ ). Assume for simplicity that  $\frac{t}{\delta}$  is an integer. We construct the graph as follows (refer to Figure 3 for an illustration).

- Let  $U = \{u_1, \dots, u_n\}$  be the  $n$  offline nodes,  $V_P = \{v_1, \dots, v_{n-d}\}$  be the  $n - d$  predicted online nodes and  $V_A = \{v_{n-d+1}, \dots, v_n\}$  be the  $d$  adversarial online nodes.
- Let the predicted graph  $H$  be a complete bipartite graph between  $U$  and  $V_P$ .
- Pick  $d$  nodes uniformly at random from  $U$  to be neighbors of  $V_A$ . Without loss of generality, let these nodes be  $U_A = \{u_{n-d+1}, \dots, u_n\}$ . Partition the  $d$  nodes in each of  $U_A$  and  $V_A$  in blocks of  $\frac{t}{\delta}$  consecutive nodes. Let  $U_A^k = \{u_{n-d+(k-1)\frac{t}{\delta}+1}, \dots, u_{n-d+(k)\frac{t}{\delta}}\}$  and  $V_A^k = \{v_{n-d+(k-1)\frac{t}{\delta}+1}, \dots, v_{n-d+(k)\frac{t}{\delta}}\}$  denote the  $k^{\text{th}}$  blocks of offline and online nodes respectively. For each  $j \leq k$ , connect all online nodes in  $V_A^j$  to all offline nodes in  $U_A^k$ . Notice that the adjacency matrix on this part of the graph looks like a block upper triangular matrix.
- Finally, the online nodes arrive in order, i.e.  $v_i$  arrives before  $v_j$  whenever  $i < j$ .

### 6.2 Analysis

After the first  $n - d$  nodes have arrived, any online algorithm can do no better than guess which offline nodes to leave unmatched uniformly at random. Let  $\tilde{d} \geq d$  be the number of offline nodes left unmatched by the best online algorithm after the arrival of all  $n - d$  predicted nodes. So at this point, we are left with a bipartite graph with  $d$  adversarial online nodes and  $\tilde{d}$  offline nodes such that each of the  $n$  total offline nodes is available with probability  $\tilde{d}/n = \tilde{\delta}$ .

Consider a block  $U_A^k$  of offline nodes. Since each node is available with probability  $\tilde{\delta}$ , in expectation  $\tilde{t} = \binom{\tilde{\delta}}{\delta} t$  nodes from the block remain available. Further, since nodes are chosen to remain available using sampling without replacement, we can obtain tight concentration

around  $\tilde{t}$ . In particular, if  $t_k$  denotes the number of available nodes remaining in block  $U_A^k$ , by Hoeffding bounds we obtain that  $\Pr(|t_k - \tilde{t}| \geq \tilde{t}^{2/3}) \leq 2e^{-\delta \cdot \tilde{t}^{1/3}}$ . Hence by a union bound over the  $\frac{\delta^2 n}{t}$  blocks, we have that *every* block has  $\tilde{t} \pm \tilde{t}^{2/3}$  available nodes with high probability (as  $t \rightarrow \infty$ ). At this point, it is somewhat clearer why blocks were chosen. Had we used single edges (as in the construction of [13]), many of them would have become unavailable, making the analysis difficult.

Let  $G'$  denote the remaining graph, that is, the graph with  $d$  adversarial online nodes and  $\tilde{d}$  remaining offline nodes. At this point, we'll analyze the water-filling algorithm [9] on  $G'$ . By [9], this is the best deterministic algorithm for *fractional* matching in the adversarial setting. Further, a lower bound on the performance of this algorithm provides a lower bound for any randomized algorithm for integer matchings.

► **Lemma 13.** *The water-filling algorithm achieves a fractional matching of total weight at most  $\delta n \cdot (1 - e^{-\delta(1+o(1))} + o(1))$  on the graph  $G'$ .*

**Proof.** Recall that in the water-filling algorithm, for each arriving online node, we spread its total weight of 1 across its incident edges so that the total fractional matching across the adjacent offline nodes is as even as possible. Let  $B = \frac{d}{t/\delta} = \frac{\delta^2 n}{t}$  be the number of blocks.

For simplicity, let's first assume that each block has exactly  $\tilde{t}$  available nodes, rather than  $\tilde{t} \pm o(\tilde{t})$ . By construction, each online node in the first block is connected to  $B\tilde{t}$  available nodes. Every online node in the second block is connected to  $(B-1)\tilde{t}$  available nodes, and so on, with every online node in the  $k$ -th block connected to  $(B-k+1)\tilde{t}$  available nodes. Hence, in the water-filling algorithm, for each of the first  $t/\delta$  online nodes, we will give  $1/(B\tilde{t})$  weight to every available offline node. Then we will give a weight of  $1/(B\tilde{t} - \tilde{t})$  to every available offline neighbor for each of the next  $t/\delta$  online nodes and so on. This process continues until the weight we have given to the last available offline node is 1, at which point we cannot allocate any more weight.

Consider the weight given to last available offline node. After seeing the first  $k+1$  blocks, this is

$$\frac{t}{\delta} \left( \frac{1}{\tilde{t}B} + \frac{1}{\tilde{t}(B-1)} + \dots + \frac{1}{\tilde{t}(B-k)} \right) \geq (1/\delta)(t/\tilde{t}) \cdot \int_{B-k+1}^{B+1} \frac{1}{x} dx = (1/\delta)(t/\tilde{t}) \ln \left( \frac{B+1}{B-k+1} \right)$$

In our case, the number of available offline nodes in each block is between  $\tilde{t} - \tilde{t}^{2/3}$  and  $\tilde{t} + \tilde{t}^{2/3}$ , w.h.p. So the amount of weight assigned to the last available node after block  $k+1$  is at least

$$\ln \left( \frac{B+1}{B-k+1} \right) \left( \frac{1}{\delta} \right) \left( \frac{t}{\tilde{t} + \tilde{t}^{2/3}} \right) \geq \ln \left( \frac{B+1}{B-k+1} \right) \left( \frac{1}{\delta} \right) \left( \frac{1}{(1 + \tilde{t}^{-1/3})} \right)$$

Note that once we have given a total weight of 1 to the last node, the water-filling algorithm will not be able to distribute any more weight. Hence, the water-filling algorithm stops after  $k$  blocks, with  $k$  being at most the smallest integer satisfying

$$\ln \left( \frac{B+1}{B-k+1} \right) \left( \frac{1}{\delta} \right) \left( \frac{1}{(1 + \tilde{t}^{-1/3})} \right) \geq 1$$

In this case,

$$k = (B+1)(1 - e^{-\delta(1 + \tilde{t}^{-1/3})}) = (B+1)(1 - e^{-\delta(1+o(1))})$$

Since each block allocates a total weight of  $\frac{t}{\delta}$ , the total weight of the fractional matching obtained by the water-filling algorithm is at most

$$\begin{aligned} \left(\frac{t}{\delta}\right) (B+1) \left(1 - e^{-\tilde{\delta}(1+o(1))}\right) &= \left(\frac{t}{\delta}\right) \left(\frac{\delta^2 n}{t} + 1\right) \left(1 - e^{-\tilde{\delta}(1+o(1))}\right) \\ &= \delta n \left(1 - e^{-\tilde{\delta}(1+o(1))}\right) \left(1 + \frac{t}{\delta^2 n}\right) \end{aligned}$$

Since, by construction, we have  $t = o(d)$ , this is  $\delta n \left(1 - e^{-\tilde{\delta}(1+o(1))} + o(1)\right)$  as desired. ◀

► **Theorem 14.** *No (randomized) algorithm for the semi-online bipartite matching problem can achieve a competitive ratio better than  $1 - \delta e^{-\delta}$ .*

**Proof.** Lemma 13 shows that after matching  $n - \tilde{d}$  predicted vertices, the best randomized algorithm can match at most  $\delta n \left(1 - e^{-\tilde{\delta}(1+o(1))} + o(1)\right)$  of the adversarial vertices. Let  $M$  be the matching found by any randomized algorithm on the graph  $G$ . Hence, we have

$$\begin{aligned} \mathbb{E}[|M|] &\leq n - \tilde{d} + \delta n \left(1 - e^{-\tilde{\delta}(1+o(1))} + o(1)\right) \leq n - \delta n + \delta n \left(1 - e^{-\tilde{\delta}(1+o(1))} + o(1)\right) \\ &= n - \delta n e^{-\tilde{\delta}(1+o(1))} + \delta n o(1) \\ &= n \left(1 - \delta e^{-\tilde{\delta}(1+o(1))} + \delta o(1)\right) \end{aligned}$$

Since  $G$  has a perfect matching of size  $n$ , the competitive ratio is upper bounded by  $(1 - \delta e^{-\delta})$  as  $n \rightarrow \infty$ . ◀

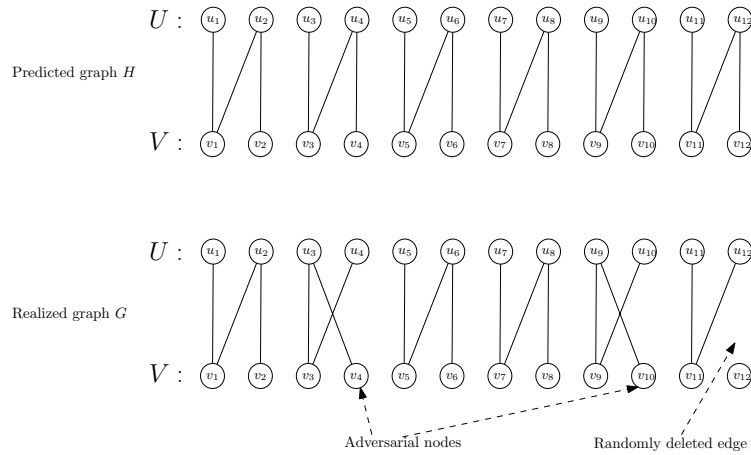
## 7 Extensions - Imperfect Predictions and Agnosticism

In this section, we consider a more general model where we allow the predicted graph to have small random errors. We define the  $(d, \epsilon)$  semi-online model as follows - We are given a predicted graph  $H = (U, V, E_H)$ , where  $|U| = |V| = n$ . As before,  $U$  are the offline nodes and  $V$  are the online<sup>5</sup> nodes. However, we do not explicitly separate  $V$  into predicted and adversarial nodes; all nodes are seen by the offline preprocessing stage, but some subset of these nodes will be altered adversarially.

An adversary selects up to  $d$  online nodes and may arbitrarily change their neighborhoods. In addition, we allow the realized graph  $G$  to introduce small random changes to the remaining predicted graph after the adversary has made its choices. Specifically, each edge in  $H$  not controlled by the adversary is removed independently with probability  $\epsilon$ . Further, for each  $u \in U, v \in V$ , we add edge  $(u, v)$  (if it does not already exist in the graph) independently with probability  $\epsilon|M|/n^2$ , where  $M$  is a maximum matching in  $H$ . Note that in expectation, we will add fewer than  $\epsilon|M|$  edges; simply adding edges with probability  $\epsilon$  (instead of  $\epsilon|M|/n^2$ ) would overwhelm the embedded matching. We call an algorithm *agnostic* if it does not know the  $d$  nodes chosen by the adversary during the preprocessing (offline) phase. There are two variants - either the algorithm knows the value of  $d$  or it does not. We show a hardness result in the former case and consider algorithms in the latter case.

We first consider agnostic algorithms to find integral matchings in this  $(d, \epsilon)$  semi-online model and give a hardness result and a corresponding tight algorithm for the case when  $\epsilon = 0$ .

<sup>5</sup> The algorithm does not know the arrival order of nodes in  $V$ .



■ **Figure 4** Hard instance for Agnostic algorithms.

► **Theorem 15.** *In the  $(d, \epsilon)$  semi-online model with  $d < n/4$ , no (randomized) agnostic algorithm can find a matching of size more than  $n - d - \epsilon(n - 3d) + O(\epsilon^2 n)$  in expectation, taken over the randomness of the algorithm and the randomness of the realized graph. This holds even if  $d$  is known in advance by the algorithm.*

**Proof.** Assume  $n$  is even. Our hard instance consists of the following predicted graph  $H$ : For each integer  $i \in [0, \frac{n}{2})$ , add edges  $(v_{2i+1}, u_{2i+1})$ ,  $(v_{2i+1}, u_{2i+2})$ , and  $(v_{2i+2}, u_{2i+2})$ . This creates  $n/2$  connected components. See Figure 4 for an illustration.

The adversary chooses  $d$  components uniformly at random. Let  $\mathcal{A} = \{i_1, i_2, \dots, i_d\} \subset [0, \frac{n}{2})$  denote the indices of the  $d$  components selected by the adversary. For each index  $i \in \mathcal{A}$ , the adversary then selects  $v_{2i+2}$  and changes its neighborhood so it only connects with  $u_{2i+1}$  (instead of  $u_{2i+2}$ ).

For simplicity, let's first consider the case when  $\epsilon = 0$ . The algorithm can do no better than picking some  $p \in [0, 1]$ , and matching  $v_{2i+1}$  to  $u_{2i+1}$  with probability  $p$ , and matching  $v_{2i+1}$  to  $u_{2i+2}$  otherwise, for all  $i$ . The algorithm then matches  $v_{2i+2}$  to its neighbor, if possible. Now, for all  $i \in \mathcal{A}$  (components selected by the adversary), this gets an expected matching of size  $p + 2(1 - p) = 2 - p$ . On the other hand, for all  $i \notin \mathcal{A}$ , the expected matching is size  $2p + (1 - p) = 1 + p$ . Since there are  $d$  components with an adversary and  $n/2 - d$  components without, this gives a total matching of size  $(2 - p)d + (1 + p)(n/2 - d) = n/2 + d + p(\frac{n}{2} - 2d)$ . This is maximized when  $p = 1$  (since  $d < n/4$ ) to yield a matching of size  $n - d$ .

When  $\epsilon > 0$ , the algorithm still should set  $p = 1$ ; if the desired edge is removed, then the algorithm will match with whatever node is available. Components with an adversarial node *gain* an edge in the matching when the edge  $(v_{2i+1}, u_{2i+1})$  is removed since the algorithm is forced into the right choice; if both edges  $(v_{2i+1}, u_{2i+1})$  and  $(v_{2i+1}, u_{2i+2})$  are removed, we neither gain nor lose. The expected gain is  $\epsilon - \epsilon^2$ . Components without an adversarial node lose an edge in the matching whenever either edge  $(v_{2i+1}, u_{2i+1})$  or edge  $(v_{2i+2}, u_{2i+2})$  is removed, and they lose an additional edge if all three edges of the component are removed. So the expected loss is  $2\epsilon - \epsilon^2 + \epsilon^3$ . Since there are  $d$  components with adversarial nodes and  $n/2 - d$  without, this is a total of loss of

$$-d(\epsilon - \epsilon^2) + (n/2 - d)(2\epsilon - \epsilon^2 + \epsilon^3) = \epsilon(n - 3d) - O(\epsilon^2 n)$$

Hence, the total matching is size  $n - \epsilon(n - 3d) + O(\epsilon^2 n)$ , as claimed. ◀

► **Theorem 16.** *Given a predicted graph  $H$  with a perfect matching, suppose there are  $d$  adversarial nodes and  $\epsilon = 0$  as described above in the  $(d, \epsilon)$  semi-online model. Then there is an agnostic algorithm that does not know  $d$  that finds a matching of expected size  $n - d$ .*

**Proof.** Before any online nodes arrive, find a perfect matching  $M$  in  $H$ . In the online stage, as each node  $v$  arrives, we attempt to identify  $v$  with an online node in the predicted graph with the same neighborhood, and match  $v$  according to  $M$ . If no node in the predicted graph has neighborhood identical to  $v$ , we know that  $v$  is adversarial and we can simply leave it unmatched. (Note that adversarial nodes can mimic non-adversarial nodes, but it doesn't actually hurt us since they are isomorphic.) The predicted matching had size  $n$ , and we lose one edge for each adversarial node, so the obtained matching has size  $n - d$ . ◀

## 7.1 Fractional matchings for predictions with errors

In this section, we show that we can find an almost optimal *fractional* matching for the  $(d, \epsilon)$  semi-online matching problem.

We use a result from [22], which gives a method of reconstructing a fractional matching using only the local structure of the graph and a single stored value for each offline node. They provide the notion of a *reconstruction function*. Their results extend to a variety of linear constraints and convex objectives, but here we need only a simple reconstruction function. For any positive integer  $k$ , define  $g^k : (\mathbb{R}_0^+)^k \rightarrow (\mathbb{R}_0^+)^k$  by

$$g^k(\alpha_1, \alpha_2, \dots, \alpha_k) = (\alpha_1 - \max(0, z), \alpha_2 - \max(0, z), \dots, \alpha_k - \max(0, z))$$

where  $z$  is a solution to  $\sum_j \min(\max(0, \alpha_j - z), 1) = 1$ .

The reconstruction function  $g$  is this family of functions. Note that this is well-defined: there is always a solution  $z$  between  $-1$  and the largest  $\alpha_j$ , and the solution is unique unless  $z \leq 0$ .

The result of [22] assigns a value  $\alpha_u$  to each  $u$  in the set of offline nodes, and reconstructs a matching on the fly as each online node arrives, using only the neighborhood of the online node and the stored  $\alpha$  values. Crucially, the reconstruction assigns reasonable values even when the neighborhood is different than predicted. In this way, it is robust to small changes in the graph structure.

► **Lemma 17** (Restated from [22]). *Let  $g_i^k$  be defined as above, and let  $H = (U, V, E_H)$  be a bipartite graph with a perfect matching of size  $n = |U| = |V|$ . Then there exist values  $\alpha_u$  for each  $u \in U$  (which can be found in polynomial time) such that the following holds: For all  $v \in V$ , define  $x_{u_i, v} = g_i(\alpha_{u_1}, \alpha_{u_2}, \dots, \alpha_{u_k})$ , where  $u_1, u_2, \dots, u_k$  is the neighborhood of  $v$ . Then  $x$  defines a fractional matching on  $H$  with weight  $n$ .*

Interested readers can find the proof in the full version. Given this reconstruction technique, we can now describe the algorithm:

- In the preprocessing phase, find the  $\alpha_u$  values for all  $u \in U$  using Lemma 17.
- In the online phase, for each online node  $v$ , compute  $\tilde{x}_{u_i, v} = g_i(\alpha_{u_1}, \dots, \alpha_{u_k})$ , where  $u_1, \dots, u_k \in \Gamma_G(v)$ , as described above. Assign weight  $\tilde{x}_{u_i, v}$  to the edge from  $u_i$  to  $v$ ; if that would cause node  $u_i$  to have more than a total weight 1 assigned to it, just assign as much as possible.

Note that we make the online computation based on the neighborhood in  $G$ , the realized graph, although the  $\alpha_u$  values were computed based on  $H$ , the predicted graph. We have the following.

► **Theorem 18.** *In the  $(d, \epsilon)$  semi-online matching problem in which the predicted graph has a perfect matching, there is a deterministic agnostic algorithm that gives a fractional matching of size  $n(1 - 2\epsilon - \delta)$  in expectation, taken over the randomized realization of the graph. The algorithm does not know the value of  $d$  or the value of  $\epsilon$  in advance.*

**Proof.** If the realized graph were exactly as predicted, we would give the fractional assignment  $x$  guaranteed in Lemma 17, which has weight  $n$ . However, the fractional matching that is actually realized is somewhat different. For each online node that arrives, we treat it the same whether it is adversarial or not. But we have a few cases to consider for analysis:

- Case 1: The online node  $v$  is adversarial. In this case, we forfeit the entire weight of 1 in the matching. We may assign some fractional matching to incident edges. However, we count this as ‘excess’ and do not credit it towards our total. In this way, we lose at most  $\delta n$  total weight.
- Case 2: The online node  $v$  is not adversarial, but it has extra edges added through a random process. There are at most  $\epsilon n$  such nodes in expectation. In this case, we treat them the same as adversarial. We forfeit the entire weight of 1, and ignore the ‘excess’ assignment. This loses at most  $\epsilon n$  total weight in expectation.
- Case 3: The online node  $v$  is as exactly as predicted. In this case, we correctly calculate  $x_{uv}$  for each  $u \in \Gamma(v)$ . Further, we assign  $x_{uv}$  to each edge, unless there was already ‘excess’ there. Since we never took credit for this excess, we will take  $x_{uv}$  credit now. So we do not lose anything in this case.
- Case 4: The online node  $v$  is as predicted, except each edge is removed with probability  $\epsilon$  (and no edges are added). In this case, when we solve for  $z$ , we find a value that is bounded above by the true  $z$ . The reason is that in the predicted graph, we solved  $\sum_{u \in \Gamma(v)} \min(\max(0, \alpha_u - z), 1) = 1$  for  $z$  when computing  $g$ . In the realized graph, this same sum has had some of its summands removed, meaning the solution in  $z$  is at most what it was before. So the value of  $\tilde{x}_{uv}$  that we calculate is at least  $x_{uv}$  for all  $u$  in the realized neighborhood. We take a credit of  $x_{uv}$  for each of these, leaving the rest as excess. Note that we have assigned 0 to each edge that was in the predicted graph but missing in the realized graph. Since each edge goes missing with probability  $\epsilon$ , this is a total of at most  $\epsilon n$  in expectation.

So, the total amount we lose in expectation is  $2\epsilon n + \delta n$ . Since the matching in the predicted graph has weight  $n$ , the claim follows. ◀

## 7.2 Semi-Online Algorithms For Ski Rental

In this section, we consider the semi-online ski rental problem. In the classical ski rental problem, a skier needs to ski for an unknown number of days and on each day needs to decide whether to rent skis for the day at a cost of 1 unit, or whether to buy skis for a higher cost of  $b$  units and ski for free thereafter. We consider a model where the skier has perfect predictions about whether or not she will ski on a given day for a few days in the time horizon. In addition, she may or may not ski on the other days. For instance, say the skier knows whether or not she’s skiing for all weekends in the season, but is uncertain of the other days. The goal is to design an algorithm for buying skis so that the total cost of skiing is competitive with respect to the optimal solution for adversarial choices for all the days for which we have no predictions.

Let  $x$  denote the number of days that the predictions guarantee the skier would ski. Further, it is more convenient to work with the fractional version of the problem so that it costs 1 unit to buy skis and renting for  $z$  (fractional) days costs  $z$  units. In this setting, we



know in advance that the skier will ski for at least  $x$  days. There is a randomized algorithm that guarantees a competitive ratio of  $1/(1 - (1 - x)e^{-(1-x)})$ . Our analysis is a minor extension of an elegant result of [11].

► **Theorem 19.** *There is a  $\frac{e}{e - (1 - x)e^x}$  competitive randomized algorithm for the semi-online ski-rental problem where  $x$  is a lower bound of the number of days the skier will ski.*

**Proof.** Without loss of generality, we can assume that all the days with a prediction occur before any of the adversarial days arrive. Otherwise, the algorithm can always pretend as if the predictions have already occurred, since only the number of skiing days is important and not their order. Recall that  $x$  denotes the number of days that the predictions guarantee the skier would ski. Let  $u \geq x$  be the actual number of days (chosen by the adversary) that she will ski. Since buying skis costs 1, the optimal solution has a cost of  $\min(u, 1)$ . Clearly, if  $x \geq 1$ , we must always buy the skis immediately and hence we assume that  $0 \leq x < 1$  in the rest of the section. Further, even the optimal deterministic algorithm buys skis once  $z = 1$ , so we may assume that  $u \leq 1$ .

Let  $p_x(z)$  denote the probability that we buy the skis on day  $z$ , and let  $q(x)$  denote the probability that we buy skis immediately. Recall that  $p_x$  is implicitly a function of the prediction  $x$ . Given a fixed number of days to ski  $u$ , we can now compute the expected cost of the algorithm as

$$\text{Cost}(x, u) = q(x) + \int_0^u (1 + z) \cdot p_x(z) dz + \int_u^1 u \cdot p_x(z) dz$$

Our goal is to choose a probability distribution  $p$  so as to minimize  $\text{Cost}(x, u)/\min(u, 1)$  while the adversary's goal is to choose  $u$  to maximize the same quantity. We will choose  $p_x$  and  $q$  so that  $\text{Cost}(x, u)/\min(u, 1)$  is constant with respect to  $u$ . As we noted,  $u \leq 1$ , so  $\min(u, 1) = u$ . Setting the  $\text{Cost}(x, u) = c \cdot u$  for constant  $c$  and taking the derivative with respect to  $u$  twice gives us

$$0 = \frac{\partial}{\partial u} p_x(u) - p_x(u)$$

Of course,  $p_x$  must also be a valid probability distribution. Thus, we set  $p_x(z) = (1 - q(x)) \cdot \frac{e^z}{e - e^x}$  for  $z \geq x$ . For  $z < x$ , we set  $p_x(z) = 0$  since there is no reason to buy skis while  $z < x$  if we did not already buy it immediately.

Recalling that we set  $\text{Cost}(x, u) = c \cdot u$ , we can substitute  $p_x(z)$  and solve for  $q(x)$ , finding

$$q(x) = \frac{xe^x}{e - (1 - x)e^x}$$

Hence, the competitive ratio is thus given by

$$\frac{\text{Cost}(x, u)}{u} = \frac{1}{u} \left( q(x) + \frac{1 - q(x)}{e - e^x} \int_x^u (1 + z)e^z dz + \frac{1 - q(x)}{e - e^x} \int_u^1 u \cdot e^z dz \right)$$

Substitute  $q(x)$ , and after some manipulation, this becomes

$$\frac{\text{Cost}(x, u)}{u} = \frac{e}{e - (1 - x)e^x}$$

Note that when  $x = 0$ , this becomes the classic ski rental problem, and the above bound is  $e/(e - 1)$ , as expected. ◀

## References

- 1 S. Albers and M Hellwig. Semi-online scheduling revisited. *Theor. Comput. Sci.*, 443:1–9, 2012.
- 2 J. Balogh and J Békési. Semi-on-line bin packing: a short overview and a new lower bound. *Cent. Eur. J. Oper. Res.*, 21(4):685–698, 2013.
- 3 Hans-Joachim Böckenhauer, Dennis Komm, Rastislav Královic, Richard Královic, and Tobias Mömke. On the Advice Complexity of Online Problems. In *ISAAC*, pages 331–340, 2009.
- 4 Allan Borodin and Ran El-Yaniv. *Online Computation and Competitive Analysis*. Cambridge University Press, 2005.
- 5 Sebastien Bubeck and Aleksandrs Slivkins. The best of both worlds: Stochastic and adversarial bandits. In *COLT*, pages 42.1–42.23, 2012.
- 6 Rajiv Gandhi, Samir Khuller, Srinivasan Parthasarathy, and Aravind Srinivasan. Dependent rounding and its applications to approximation algorithms. *Journal of the ACM (JACM)*, 53(3):324–360, 2006.
- 7 Ashish Goel, Michael Kapralov, and Sanjeev Khanna. On the communication and streaming complexity of maximum bipartite matching. In *SODA*, pages 468–485. Society for Industrial and Applied Mathematics, 2012.
- 8 Pascal Van Hentenryck and Russell Bent. *Online stochastic combinatorial optimization*. The MIT Press, 2009.
- 9 Bala Kalyanasundaram and Kirk Pruhs. An optimal deterministic algorithm for online b-matching. *Theor. Comput. Sci.*, 233(1-2):319–325, 2000. doi:10.1016/S0304-3975(99)00140-1.
- 10 Chinmay Karande, Aranyak Mehta, and Pushkar Tripathi. Online bipartite matching with unknown distributions. In *STOC*, pages 587–596, 2011.
- 11 Anna R. Karlin, Claire Kenyon, and Dana Randall. Dynamic TCP Acknowledgment and Other Stories about  $e/(e-1)$ . *Algorithmica*, 36(3):209–224, 2003.
- 12 Anna R. Karlin, Mark S. Manasse, Lyle A. McGeoch, and Susan Owicki. Competitive randomized algorithms for nonuniform problems. *Algorithmica*, 11(6):542–571, 1994.
- 13 Richard M Karp, Umesh V Vazirani, and Vijay V Vazirani. An optimal algorithm for on-line bipartite matching. In *STOC*, pages 352–358, 1990.
- 14 Ravi Kumar, Manish Purohit, and Zoya Svitkina. Improving Online Algorithms Using ML Predictions. In *NIPS*, 2018.
- 15 Euiwoong Lee and Sahil Singla. Maximum Matching in the Online Batch-Arrival Model. In *IPCO*, pages 355–367, 2017.
- 16 Thodoris Lykouris and Sergei Vassilvitskii. Competitive caching with machine learned advice. In *ICML*, pages 3302–3311, 2018.
- 17 Mohammad Mahdian, Hamid Nazerzadeh, and Amin Saberi. Online optimization with uncertain information. *ACM Trans. Algorithms*, 8(1):2:1–2:29, 2012.
- 18 Mohammad Mahdian and Qiqi Yan. Online bipartite matching with random arrivals: an approach based on strongly factor-revealing LPs. In *STOC*, pages 597–606, 2011.
- 19 Andres Muñoz Medina and Sergei Vassilvitskii. Revenue Optimization with Approximate Bid Predictions. In *NIPS*, pages 1856–1864, 2017.
- 20 Aranyak Mehta. Online Matching and Ad Allocation. *Foundations and Trends® in Theoretical Computer Science*, 8(4):265–368, 2013.
- 21 Vahab S. Mirrokni, Shayan Oveis Gharan, and Morteza Zadimoghaddam. Simultaneous approximations for adversarial and stochastic online budgeted allocation. In *SODA*, pages 1690–1701, 2012.
- 22 Erik Vee, Sergei Vassilvitskii, and Jayavel Shanmugasundaram. Optimal online assignment with forecasts. In *EC*, pages 109–118, 2010.

# Strategies for Quantum Races

## Troy Lee

Centre for Quantum Software and Information, School of Software,  
Faculty of Engineering and Information Technology, University of Technology Sydney, Australia  
troyjlee@gmail.com

## Maharshi Ray

Centre for Quantum Technologies, National University of Singapore, Singapore  
maharshi91@gmail.com

## Miklos Santha

IRIF, Univ. Paris Diderot, CNRS, 75205 Paris, France; and  
Centre for Quantum Technologies and MajuLab,  
National University of Singapore, Singapore 117543  
miklos.santha@gmail.com

---

### Abstract

---

We initiate the study of *quantum races*, games where two or more quantum computers compete to solve a computational problem. While the problem of dueling algorithms has been studied for classical deterministic algorithms [12], the quantum case presents additional sources of uncertainty for the players. The foremost among these is that players do not know if they have solved the problem until they measure their quantum state. This question of “when to measure?” presents a very interesting strategic problem. We develop a game-theoretic model of a multi-player quantum race, and find an approximate Nash equilibrium where all players play the same strategy. In the two-party case, we further show that this strategy is nearly optimal in terms of payoff among all symmetric Nash equilibria. A key role in our analysis of quantum races is played by a more tractable version of the game where there is no payout on a tie; for such races we completely characterize the Nash equilibria in the two-party case.

One application of our results is to the stability of the Bitcoin protocol when mining is done by quantum computers. Bitcoin mining is a race to solve a computational search problem, with the winner gaining the right to create a new block. Our results inform the strategies that eventual quantum miners should use, and also indicate that the collision probability – the probability that two miners find a new block at the same time – would not be too high in the case of quantum miners. Such collisions are undesirable as they lead to forking of the Bitcoin blockchain.

**2012 ACM Subject Classification** Theory of computation → Algorithmic game theory

**Keywords and phrases** Game theory, Bitcoin mining, Quantum computing, Convex optimization

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2019.51

**Related Version** A full version of the paper is available at [14], <https://arxiv.org/abs/1809.03671>.

**Acknowledgements** We would like to thank Gavin Brennen, Hartmut Klauck, and Marco Tomamichel for very useful discussions about game theory and quantum Bitcoin mining. We would also like to thank Or Sattath for sharing an early version of his work [18] and discussions about its implications.

Part of this work was done while Troy Lee was at the School of Physical and Mathematical Sciences, Nanyang Technological University and the Centre for Quantum Technologies, Singapore. Troy Lee and Maharshi Ray were supported in part by the Singapore National Research



© Troy Lee, Maharshi Ray, and Miklos Santha;  
licensed under Creative Commons License CC-BY  
10th Innovations in Theoretical Computer Science (ITCS 2019).  
Editor: Avrim Blum; Article No. 51; pp. 51:1–51:21



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

Fellowship under NRF RF Award No. NRF-NRFF2013-13. This research was further partially funded by the Singapore Ministry of Education and the National Research Foundation under grant R-710-000-012-135, and in part by the QuantERA ERA-NET Cofund project QuantAlgo.

## 1 Introduction

We study the scenario of two or more quantum computers competing to solve a computational task, which we call a *quantum race*. This setting presents a different problem to finding the fastest algorithm for a task, as the only goal is to solve the task before the competitors. For example, imagine a search race where Alice and Bob, each armed with identical quantum computers, compete to find a marked item in a database. The first person to find the marked item wins \$1, with the payout being split in the case of a tie. The first natural idea is for Alice to run Grover’s algorithm [11], which can find a marked item in a database of size  $N$  with high probability in time  $O(\sqrt{N})$ . However, if Alice’s strategy is to run Grover’s algorithm and measure after the specified number of steps to maximize her success probability, Bob will have an advantage by measuring after running Grover’s algorithm for a few less steps. Although this way Bob has a slightly lower success probability, he gains a huge advantage in always answering first. This simple example shows that the optimal algorithm to solve a problem can be different from the optimal strategy to employ when the goal is to solve the problem before an opponent.

The scenario of competing algorithms has been studied before in the classical deterministic setting [12]. In a classical game, the uncertainty is provided by an unknown probability distribution over the inputs: depending on what the input is, one algorithm may perform better than another. The quantum setting inherently has additional sources of uncertainty, most interestingly that players do not know if they have solved the problem until they *measure* their quantum state. Going back to the search game, in the classical version the players know at every instant if they have found the marked item or not. This is not the case in the quantum setting, where a player can only tell if she has found the desired item by measuring her quantum state. Furthermore, if she measures her state and does not find the marked item, then she must begin the search again from scratch. In the quantum case there is a natural tension between waiting to measure, and thereby building up the probability of success upon measuring, and measuring sooner, to answer before one’s competitors. We study this game theoretic problem to develop strategies for players to use in quantum races.

One of our main motivations for studying quantum races is to model quantum computers mining the decentralized currency Bitcoin [16]. Mining is the process by which new blocks of transactions are added to the history of Bitcoin transactions, called the blockchain. The winner of a race to solve a computational search problem gains the right to add a new block of transactions to the blockchain, and participants in this race are called miners. Quantum miners could use Grover’s algorithm to solve the search problem with quadratically fewer search queries than needed classically. But what should the strategy of quantum miners be when competing against each other?

► **Question 1.** *What is the optimal strategy for quantum miners?*

Figuring out the optimal strategy for quantum miners is important to analyze the impact of quantum mining on the stability of the Bitcoin protocol. When two miners solve the computational search problem at (nearly) the same time, the blockchain can fork as it is unclear which new block is the “correct” history of Bitcoin transactions. Forking is bad for the security of Bitcoin as it can decrease the cost of attacks [10], increase the gain from

deviating from the intended mining protocol [9], and generally decreases chain growth and wastes resources. In the classical case, each search query has the same probability of success. In the quantum case, however, because of Grover’s algorithm the success probability grows roughly quadratically with the number of search queries. Does this lead to many quantum miners finding blocks at the same time?

► **Question 2.** *What is the probability that two or more quantum miners playing the optimal strategy find a block at the same time?*

In the next subsections, we describe our model and results in more detail and the impact it has on these questions.

## 1.1 The model and results

In a *symmetric* game all players have the same payoff function. In his original paper defining a Nash equilibrium, Nash showed that every symmetric multiparty finite game has a symmetric equilibrium, i.e. one where all players play the same strategy [17]. When all players have identical quantum computers, a quantum race is naturally a symmetric race, and we describe this scenario first.

We model a symmetric multiplayer quantum race in the following way. The pure strategies available to a player are the possible times at which she can measure  $1, 2, 3, \dots, K$ . For each time  $t$ , a player has an algorithm that she can run for  $t$  steps and for which the success probability is  $p_t$ . Without loss of generality, we assume that these probabilities form an increasing sequence  $0 < p_1 < p_2 < \dots < p_K \leq 1$ . A general strategy is a probability distribution over the possible times to measure. The player who succeeds first receives a payoff of 1. In the case of a tie, the payoff is split amongst all players who succeed first at the same time. Our model can be thought of as a “one-shot” race, as if a player measures and does not succeed, she does not get a chance to restart and try again. While a race where players are allowed to repeatedly restart until someone wins would be more realistic, it becomes much more difficult to analyze due to the proliferation of possible strategies, and we leave this for future work.

### Two-player case

We begin explaining our model and results in more detail in the two-player case. In this case, a game defined by the probabilities  $p_1, \dots, p_K$  can be represented by the payoff matrix for Alice, given by the  $K$ -by- $K$  matrix  $A$ , and the payoff matrix for Bob  $B$ . The  $(s, t)$  entry of  $A$  gives Alice’s payoff when she runs an algorithm for time  $s$  and Bob plays time  $t$ . In the case of a quantum race, this is defined as

$$A(s, t) = \begin{cases} p_s & \text{if } s < t \\ p_s(1 - p_s) + \frac{1}{2}p_s^2 & \text{if } s = t \\ p_s(1 - p_t) & \text{if } s > t \end{cases} \quad (1)$$

As the game is symmetric, Bob’s payoff matrix is  $B = A^T$ .

Our analysis of quantum races begins in Section 3 by analyzing a more tractable variant of the game we call a *stingy* quantum race. In a stingy quantum race, there is no payout in the case of a tie (the game organizer is stingy). A Nash equilibrium of a two-party stingy quantum race has very strong constraints on its support structure (see Corollary 16). In particular, if  $(x, y)$  is a Nash equilibrium in a two-party stingy quantum race, then the union

of the supports of  $x$  and  $y$  must be an interval  $\{T, T + 1, \dots, K\}$  that contains the maximum running time  $K$ . There are 3 possible types of Nash equilibria in a two-party stingy quantum race, and we characterize all of them (see Theorem 22, and Appendix of [14]).

One particularly nice type of equilibrium is what we call a *coinciding* equilibrium. In a coinciding equilibrium, the support of all player's strategies is the same, but the strategies do not have to be identical. This is a more general notion than a symmetric equilibrium where all strategies are the same. In a coinciding equilibrium for a stingy quantum race, the support of each player's strategy is an interval  $\{T, T + 1, \dots, K\}$ . This leaves the problem of determining the starting point  $T$  of this interval in a Nash equilibrium. We are able to show that there is always exactly one  $T$  such that there is a Nash equilibrium with support  $\{T, T + 1, \dots, K\}$ .

► **Theorem 3** (Informal, see Theorem 22). *In a two-party stingy quantum race defined by probabilities  $p_1, \dots, p_K$ , there is a unique coinciding Nash equilibrium. In this equilibrium all players play the same strategy, and the support of the strategies is an interval  $\{T^*, T^* + 1, \dots, K\}$ .*

We also explicitly find this Nash equilibrium.

This result begs the question: what is this value of  $T^*$ ? At what success probability does it become worthwhile to start measuring? By putting an additional restriction on the probabilities  $p_1, \dots, p_K$ , we can give quite a precise answer to this question. We say  $p_1, \dots, p_K$  is an  $\ell$ -dense sequence (see Definition 13) if  $p_1 \leq \frac{\ell}{K}, p_K \geq 1 - \frac{\ell}{K}$ , and  $p_{i+1} - p_i \leq \frac{\ell}{K}$  for  $i = 2, \dots, K - 1$ . This is quite a natural restriction that is satisfied for many races. In the quantum search race, where the  $p_i$  are the Grover success probabilities, and therefore also in the application to Bitcoin, the  $\ell$ -density condition is satisfied with  $\ell = \pi/2$ . In the  $\ell$ -dense case, we can give the following bound on  $T^*$ .

► **Theorem 4** (Informal, see Corollary 25 and Corollary 28). *Let  $p_1, \dots, p_K$  be an  $\ell$ -dense sequence with  $K \geq 6\ell$ . Then the starting point  $T^*$  of the unique coinciding Nash equilibrium in the stingy quantum race defined by these probabilities is such that  $p_{T^*} = \sqrt{2} - 1 + \Theta\left(\frac{\ell}{K}\right)$ .*

Thus it is worthwhile to start measuring once the success probability becomes around  $\sqrt{2} - 1$ , and this is largely independent of the actual values of  $p_1, \dots, p_K$ .

In Section 4, we apply our analysis of two-player stingy quantum races to the case of general quantum races. As the only difference between a stingy quantum race and a quantum race is the payout on ties, intuitively strategies in these two kinds of races should have similar payoffs when the probability of ties is small. We follow this intuition and show that when  $p_1, \dots, p_K$  form an  $\ell$ -dense sequence the probability of a tie is  $O\left(\frac{\ell}{K}\right)$  (see Theorem 27) when players use the unique coinciding equilibrium from the stingy race, and this strategy is an  $O\left(\frac{\ell}{K}\right)$ -approximate Nash equilibrium of the corresponding quantum race.

Approximate Nash equilibria are naturally an imperfect lens into true Nash equilibria. The approximate Nash equilibrium we give would not be a reasonable suggestion for the actual strategies of quantum players if there were other equilibria with much higher payoff, for example. We show that this is not the case, and the approximate Nash equilibrium we give is nearly optimal in terms of payoff amongst all symmetric equilibria.

► **Theorem 5** (Informal, see Theorem 31 and Theorem 33). *Let  $p_1, \dots, p_k$  be an  $\ell$ -dense sequence with  $K \geq 6\ell$ . Then the unique coinciding equilibrium of the two-player stingy quantum race defined by these probabilities is an  $O\left(\frac{\ell}{K}\right)$ -approximate Nash equilibrium in the corresponding quantum race. Moreover, the payoff achieved by this strategy is within  $O\left(\sqrt{\frac{\ell}{K}}\right)$  of the largest payoff achievable by any symmetric Nash equilibrium.*



To show that the approximate Nash equilibrium we give is nearly optimal in terms of payoff (Theorem 33), we use the bilinear programming formulation of Nash equilibria due to Mangasarian and Stone [15]. We exploit the properties of the sum of Alice's and Bob's payoff matrices  $A + A^T$  (from Eq. (1)). More specifically, It turns out that over a probability simplex  $x$ , the quadratic form  $x^T(A + A^T)x$  is a negative-definite plus linear function. When optimizing over symmetric strategies this makes the Mangasarian and Stone bilinear program (which is a maximization problem) into a convex quadratic program. We then use Dorn's [8] equivalent dual formulation of a convex quadratic program (see Eq. (17)), which is a minimization problem. We explicitly construct a feasible solution to this dual minimization problem to upper bound the payoff of any symmetric Nash equilibrium. Our construction of this dual solution again makes use of our analysis of stingy quantum races.

### Multiplayer case

The case of many players is what we are interested in for the application to Bitcoin. Luckily, we are able to recover analogs of many of the results from the two-player case in the multiplayer case as well. We start in Section 5 by analyzing  $n$ -player stingy quantum races, and show the following.

► **Theorem 6** (Informal, see Theorem 41). *Let  $p_1, \dots, p_K$  define an  $n$ -player stingy quantum race. This race has a unique coinciding Nash equilibrium, and in this equilibrium all players play the same strategy. The support of each strategy is an interval  $\{T^*, T^* + 1, \dots, K\}$ .*

To show that an  $n$ -player stingy quantum race has a unique coinciding Nash equilibrium, our proof proceeds through a 2-player *asymmetric* stingy quantum race. In a 2-player asymmetric race, Alice has probabilities  $p_1, \dots, p_K$  of succeeding after  $t$  steps and Bob has a (potentially different) sequence of probabilities  $P_1, \dots, P_K$ . An asymmetric race models the case where Alice and Bob have quantum computers of potentially different speeds. We relate the payoff for Alice in a  $n$ -player stingy quantum race to the payoff for Alice in a 2-player quantum race against a more powerful opponent (see Lemma 40). We can then refer to Theorem 21 in Section 3 which completely characterizes coinciding equilibria in asymmetric 2-player stingy quantum races. This gives Theorem 6.

When the sequence  $p_1, \dots, p_K$  is  $\ell$ -dense, we can also say something about the starting point  $T^*$  of the  $n$ -player coinciding equilibrium, though not as precisely as in the two-party case.

► **Theorem 7** (Informal, see Theorem 44). *Let  $p_1, \dots, p_K$  be an  $\ell$ -dense sequence defining a stingy  $n$ -player quantum race with  $n \geq 2$ . If  $K \geq 4e\ell n$  then the starting point  $T^*$  of the unique coinciding equilibrium is such that  $p_{T^*} = \Theta(\frac{1}{n})$ .*

This means that the more players there are in a game, the earlier one starts to measure in the unique coinciding equilibrium.

In light of Question 2 we also want to see what the probability of more than one player succeeding at the same time in this unique coinciding equilibrium. We show the following.

► **Theorem 8.** *Let  $P_1, \dots, P_K$  define an  $\ell$ -dense stingy  $n$ -player quantum race such that  $4e\ell n \leq K$ . When the players play the coinciding equilibrium of the stingy race, the probability that two or more players succeed at the same time is at most  $\frac{8e\ell n}{K}$ .*

Finally, as in the two-party case, we show that the unique coinciding equilibrium of a stingy race is also an approximate Nash equilibrium in the corresponding quantum race, provided the sequence of probabilities is  $\ell$ -dense.



► **Theorem 9.** *Let  $P_1, \dots, P_K$  define an  $\ell$ -dense stingy  $n$ -player quantum race,  $n \geq 2$ , with  $4en\ell \leq K$ . If  $x = (x_1, \dots, x_n)$  is the coinciding Nash equilibrium for this stingy race, then  $x$  is an  $\frac{8e\ell}{K}$ -approximate Nash equilibrium of the corresponding quantum race.*

## 1.2 Application to Bitcoin

One application of our study of quantum races is to the decentralized digital currency Bitcoin, developed in 2008 by Satoshi Nakamoto [16]. Bitcoin transactions are packaged into blocks and stored in a public ledger called the blockchain. A major obstacle in creating a decentralized currency is to find a way for all parties to agree on the history of transactions. In Bitcoin, this is done through *Nakamoto consensus*: the right to create a new block is decided through proof-of-work, a contest to solve a computational problem. The winner of this contest has the right to make a new block of transactions, is given a reward in bitcoin, and then the process repeats itself. The players competing in this process are called miners. Nakamoto consensus remains the primary means of achieving consensus across all cryptocurrencies, although there are coins using other consensus mechanisms such as proof-of-stake [13] or Byzantine agreement [6].

The proof-of-work task used in Bitcoin (originally developed in a system called Hashcash [2]) is essentially a search problem. The problem is to find a value  $x$  (called a nonce) such that  $h(H \parallel x) \leq t$ , where  $h$  is a hash function (doubly iterated SHA-256 in the case of Bitcoin),  $H$  is the header of the block of transactions to be processed, and  $t$  is a hardness parameter that can be varied so that the entire network takes 10 minutes to solve this task, on average.

Several works have studied the impact that quantum computers would have on the Bitcoin protocol [5, 1, 18], both on the mining process we have described above and on the digital signatures used in Bitcoin to authenticate ownership of coins. We will focus here on the impact of quantum computers on Bitcoin mining.

As the Bitcoin proof-of-work is a search task, quantum miners could use Grover's algorithm to find a nonce  $x$  satisfying  $h(H \parallel x) \leq t$  with quadratically fewer evaluations of the hash function  $h$  than is needed by a classical computer<sup>1</sup>. The use of Grover's algorithm creates new issues for proof-of-work that do not exist in the classical case. Desirable properties of a proof-of-work task have been studied from an axiomatic point of view by Biryukov and Khovratovich [3]. One property they give is *progress-freeness*: the probability of a miner solving the proof-of-work task in any moment is independent of previous events. This is achieved for a classical miner in the Bitcoin proof of work, as every call to the hash function is equally likely to find a good nonce  $x$ . Progress-freeness is not achieved for a quantum miner running Grover's algorithm, as the success probability grows roughly quadratically with the amount of time the algorithm is run.

Sattath [18] points out that this gives a way for quantum miners to deviate from the prescribed protocol in order to increase their chance of winning a block. To explain this deviation, imagine a simplified case where the proof-of-work is to find a unique marked item in a database of size  $N$ . Say that Alice, a quantum miner, receives a new block from the network which was found by Bob. When Alice receives this block she will be in the middle of running Grover's algorithm to find the marked item herself. The prescribed protocol says that she should immediately halt this run of Grover's algorithm and begin working

<sup>1</sup> While this seems to give quantum computers a huge advantage for Bitcoin mining, specialized classical Bitcoin mining hardware currently can perform 14 trillion hashes per second [4] and would outperform a near-term quantum computer with gate speeds of 100MHz [1]

on a new search problem by mining on top of Bob's new block. However, if Alice has run Grover algorithm for  $c\sqrt{N}$  steps, for a constant  $c$ , she will have already built up a constant probability of finding the marked item upon measuring. From the point of view of maximizing her payoff, there is no harm in just measuring to see if she finds the marked item. If Alice gets lucky and indeed finds the marked item, then she can broadcast her new block to the network. Depending on her connectivity to the network, some other miners may receive Alice's block before Bob's, and there is some probability that Alice's new block eventually becomes the block accepted by the network rather than Bob's, meaning that Alice will receive the bitcoin reward. Note that this does not happen in the classical case, where after Alice receives Bob's block she would still just have probability  $1/N$  to find the marked item with each additional search query. In this case it makes sense to immediately start mining on top of the new block.

Luckily, Sattath also provides an easy fix for the Bitcoin protocol to remedy this problem. Without going into the technical details, this fix essentially forces miners to commit to how long they will run Grover's algorithm *before they begin*. Thus if Alice commits to running Grover's algorithm for time  $\sqrt{N}/100$ , yet receives Bob's block after time  $\sqrt{N}/200$ , if she tries to immediately measure and publish her own block, the network will reject it because of the timing discrepancy. This fix fits in very well with our model of quantum races, as a strategy is exactly a probability distribution over choices of times to measure.

The quantum race that captures the case of Bitcoin mining is what we call the *Grover race* (see Definition 12). In this race, the success probability  $p_t$  is given by the success probability of  $t$ -iterations of Grover's algorithm<sup>2</sup>. This race is an  $\ell$ -dense race for  $\ell = \pi/2$ . The size of the search space, and thus the maximum number of iterations  $K$  to run Grover's algorithm, is determined by the *difficulty* setting of the Bitcoin protocol. Currently the difficulty (as of September 7, 2018) is approximately  $7 \cdot 10^{12}$ , which, by Bitcoin's definition of difficulty, means that the network has to do roughly  $2^{32} \cdot 7 \cdot 10^{12}$  many hashes to succeed, in expectation. This leads to a value of  $K$  of approximately  $10^{11}$ . Thus for this application  $\frac{\ell}{K}$  is very small, and Theorem 9 implies that the unique coinciding equilibrium for the stingy Grover race is an  $\epsilon$ -approximate Nash equilibrium in the Grover race for  $\epsilon \leq 3 \cdot 10^{-10}$ . This gives a reasonable answer to Question 1 for what a good strategy would be for quantum miners, and moreover has the desirable property that all miners run the same algorithm. By Theorem 8, when there are  $n$  miners running this strategy the probability of a tie is at most  $3n \cdot 10^{-10}$ . This gives an answer to Question 2, that quantum mining is not likely to produce a high forking rate and thereby destabilize the Bitcoin protocol.

## 2 Preliminaries

We use  $e \approx 2.71828$  for Euler's number. For a probability  $0 \leq p \leq 1$ , we set  $\bar{p} = 1 - p$ . For a natural number  $n$ , we let  $[n] = \{1, \dots, n\}$ . We let  $\Delta_n = \{x \in \mathbb{R}^n : x \geq 0, \sum_{i=1}^n x_i = 1\}$  be the probability simplex. For  $x \in \Delta_n$  we let  $\text{supp}(x) = \{i \in [n] : x_i > 0\}$  be the support of  $x$ .

► **Definition 10** (quantum race). A 2-player quantum race is specified by two sequences of increasing probabilities  $0 < p_1 < p_2 < \dots < p_K \leq 1$ , and  $0 < P_1 < P_2 < \dots < P_K \leq 1$  for some integer  $K \geq 2$ . The set of pure strategies of both players is  $[K]$ . The  $K \times K$  payoff

<sup>2</sup> It is known that  $t$  queries of Grover's algorithm maximizes the probability of success in a search problem over all  $t$ -query quantum algorithms [7].

matrix  $A$  of Alice and  $B$  of Bob are defined as

$$A(i, j) = \begin{cases} p_i & \text{if } i < j, \\ p_i \bar{P}_j + \frac{1}{2} p_i P_i & \text{if } i = j, \\ p_i \bar{P}_j & \text{otherwise.} \end{cases} \quad B(i, j) = \begin{cases} P_j & \text{if } j < i, \\ P_j \bar{p}_i + \frac{1}{2} p_i P_i & \text{if } i = j, \\ P_j \bar{p}_i & \text{otherwise.} \end{cases}$$

If  $p_i = P_i$  for all  $i = 1, \dots, K$  then we call the game a *symmetric* quantum race. Note that in this case  $B = A^T$ .

In our study of quantum races, a key role will be played by an auxiliary game that is easier to analyze called a stingy quantum race. A stingy quantum race differs from a quantum race only in that no payout is given in the case of a tie.

► **Definition 11** (stingy quantum race). A 2-player stingy quantum race is specified by two sequences of increasing probabilities  $0 < p_1 < p_2 < \dots < p_K \leq 1$ , and  $0 < P_1 < P_2 < \dots < P_K \leq 1$  for some integer  $K \geq 2$ . The set of pure strategies of both players is  $[K]$ . The  $K \times K$  payoff matrix  $A_0$  of Alice is defined as

$$A_0(i, j) = \begin{cases} p_i & \text{if } i < j, \\ p_i \bar{P}_j & \text{otherwise.} \end{cases}$$

The payoff matrix of Bob  $B_0$  is defined as

$$B_0(i, j) = \begin{cases} P_j & \text{if } j < i, \\ P_j \bar{p}_i & \text{otherwise.} \end{cases}$$

If  $p_i = P_i$  for all  $i = 1, \dots, K$  then we call the game a *symmetric* stingy quantum race. Note that in this case  $B_0 = A_0^T$ .

The main specific quantum race we will be interested in is the *Grover race*. This results from two players competing to find a marked item in a database and playing by running Grover's algorithm for a certain amount of time and then measuring. Formally, the race is defined as follows.

► **Definition 12** (Grover race). We define the (stingy) Grover race on  $N$  items as the symmetric (stingy) quantum race with  $K = \lceil \frac{\pi}{4} \sqrt{N} - 3/2 \rceil$  and  $p_t = \sin^2 \left( 2(t + 1/2) \arcsin \left( \frac{1}{\sqrt{N}} \right) \right)$ , for  $1 \leq t \leq K$ .

Here  $p_t$  is the success probability of Grover's algorithm of finding a unique marked item in a database of  $N$  items. It is known that  $p_t$  is the highest success probability for finding a marked item for any quantum algorithm making  $t$  many calls to the database [7].

The Grover race has many nice properties, and we will abstract out one of them here. This allows us to show results for a general class of quantum races, rather than just the Grover race.

► **Definition 13** (dense race). Let  $p_1 < p_2 < \dots < p_K \leq 1$ . We call the sequence  $(p_1, \dots, p_K)$   $\ell$ -dense if  $p_1 \leq \frac{\ell}{K}$ ,  $p_K \geq 1 - \frac{\ell}{K}$ , and  $p_{i+1} - p_i \leq \frac{\ell}{K}$  for all  $i = 1, \dots, K - 1$ . For a dense sequence  $(p_1, \dots, p_K)$  will similarly call the symmetric (stingy) quantum race defined by this sequence a symmetric (stingy)  $\ell$ -dense quantum race.

The (stingy) Grover race is an  $\ell$ -dense race with  $\ell = \pi/2$ .

### 3 Two-player stingy quantum races

We will first analyze Nash equilibria in stingy quantum races. We can show several structural properties about the support of Nash equilibria in stingy quantum races that make them easier to analyze than quantum races. After our analysis in this section, we will bootstrap our knowledge of Nash equilibria in stingy quantum races to find an approximate Nash equilibrium for quantum races.

Consider a stingy quantum race given by the probabilities  $0 < p_1 < p_2 < \dots < p_K \leq 1$  and  $0 < P_1 < P_2 < \dots < P_K$ , and let  $y$  be a mixed strategy of Bob. For  $t \leq K$ , we let  $\text{sup}_{\leq t}(y) = \{j \in \text{sup}(y) : j \leq t\}$  be the set of times played by Bob that are at most  $t$ , and similarly, we let  $\text{sup}_{> t}(y) = \{j \in \text{sup}(y) : j > t\}$  be the set of times played by Bob that are greater than  $t$ . Observe that when Alice plays the pure strategy  $t$  against  $y$ , her payoff is

$$e_t^T A y = p_t \left( \sum_{j \in \text{sup}_{\leq t}(y)} y_j \bar{P}_j + \sum_{j \in \text{sup}_{> t}(y)} y_j \right).$$

► **Claim 14.** *Let  $(x, y)$  be a Nash equilibrium of a 2-player stingy quantum race. If  $t_1 \in \text{sup}(x)$  then there does not exist  $t_2 > t_1$  with  $\text{sup}_{\leq t_1}(y) = \text{sup}_{\leq t_2}(y)$ .*

**Proof.** Say that the game is defined by probabilities  $0 < p_1 < p_2 < \dots < p_K \leq 1$  and  $0 < P_1 < P_2 < \dots < P_K$ . If  $\text{sup}_{\leq t_1}(y) = \text{sup}_{\leq t_2}(y)$  then

$$e_{t_2}^T A y = \frac{p_{t_2}}{p_{t_1}} e_{t_1}^T A y.$$

As  $p_{t_1} < p_{t_2}$ , the payoff for playing  $t_2$  is strictly larger than that for playing  $t_1$ . Therefore  $t_1$  is not a best response for  $y$ , in contradiction with the definition of a Nash equilibrium. ◀

This claim implies that the support structure of Nash equilibria in stingy quantum races is relatively simple. We first make the following definition.

► **Definition 15.** A pair of strategies  $(x, y)$  is called *coinciding* if  $\text{sup}(x) = \text{sup}(y)$ . A pair of strategies  $(x, y)$  is called *alternating* if there exists  $1 \leq t_1 < t_2 \leq K$  such that the support of one player is  $\{t_1, t_1 + 2, \dots, t_2 - 1\}$  and the support of the other is  $\text{sup}(y) = \{t_1 + 1, t_1 + 3, \dots, t_2\}$ . A pair of strategies  $(x, y)$  is called  $(t_1, c, t_2)$ -*alternating-coinciding* if there are natural numbers  $1 \leq t_1 < t_2 \leq K$  and  $t_1 + 2 \leq c \leq t_2$  such that the support of one player is  $\{t_1, t_1 + 2, \dots, c - 2, c, c + 1, c + 2, \dots, t_2\}$  and the support of the other is  $\text{sup}(y) = \{t_1 + 1, t_1 + 3, \dots, c - 3, c - 1, c, c + 1, c + 2, \dots, t_2\}$ .

► **Corollary 16.** *Let  $(x, y)$  be a Nash equilibrium of a stingy quantum race specified by probabilities  $0 < p_1 < p_2 < \dots < p_K \leq 1$  and  $0 < P_1 < P_2 < \dots < P_K$ . Then there is some  $1 \leq T \leq K$  such that*

- $\text{sup}(x) \cup \text{sup}(y) = [T, K]$  ,
- $(x, y)$  is either coinciding, alternating, or alternating-coinciding.

**Proof.** From Claim 14 we can easily derive the following two statements:

- $\text{sup}(x) \cup \text{sup}(y)$  is an interval containing the time with maximum success probability,
- For every  $t_1, t_2 \in \text{sup}(x)$  there must be  $t \in \text{sup}(y)$  with  $t_1 < t \leq t_2$ .

These statements immediately imply the claim. ◀

We now study the particularly simple coinciding equilibria. Due to space constraints we do not discuss the other types of equilibria here. However, we give a full characterization of all Nash equilibria in a symmetric stingy quantum race in the full version of the paper [14].

### 3.1 Unique coinciding equilibrium

We first look for Nash equilibria where the mixed strategies of Alice and Bob have the same support. If the number of strategies is  $K$ , we know by Corollary 16 that this support must be a set  $\{T, T+1, \dots, K\}$ , for some  $1 \leq T \leq K$ .

► **Lemma 17.** *Consider a stingy quantum race defined by  $0 < p_1 < \dots < p_K \leq 1$  and  $0 < P_1 < P_2 < \dots < P_K$ . Let  $x, y \in \mathbb{R}^K$ . Then  $(x, y)$  is a Nash equilibrium for this game with support  $\{T, T+1, \dots, K\}$ , for some  $1 \leq T \leq K$ , if and only if  $x$  and  $y$  satisfy the following system of equations and inequalities.*

$$e_t^\top Ay = e_T^\top Ay, \quad \text{for } T < t \leq K, \quad (2)$$

$$e_{T-1}^\top Ay \leq e_T^\top Ay, \quad (3)$$

$$y_t = 0 \quad \text{for } 0 < t < T, \quad (4)$$

$$y_t > 0 \quad \text{for } T \leq t \leq K, \quad (5)$$

$$\sum_{t=T}^K y_t = 1 \quad (6)$$

$$x^\top Be_t = x^\top Be_T, \quad \text{for } T < t \leq K, \quad (7)$$

$$x^\top Be_{T-1} \leq x^\top Be_T, \quad (8)$$

$$x_t = 0 \quad \text{for } 0 < t < T, \quad (9)$$

$$x_t > 0 \quad \text{for } T \leq t \leq K, \quad (10)$$

$$\sum_{t=T}^K x_t = 1. \quad (11)$$

**Proof.** Eq. (4)–(6) and (9)–(11) express that  $x$  and  $y$  are probability distributions with support  $\{T, T+1, \dots, K\}$ . The other conditions for a Nash equilibrium are that all strategies in the support of  $x$  are best responses against  $y$  and vice versa. That all strategies in the support of  $x$  are best responses against  $y$  means

$$e_t^\top Ay = e_T^\top Ay, \quad \text{for } T < t \leq K,$$

$$e_{t-1}^\top Ay \leq e_t^\top Ay, \quad \text{for } 1 \leq t < T.$$

The first equation here is exactly Eq. (2). The inequality here is implied by Eq. (3). This is because for  $t < T-1$ ,  $e_t^\top Ay = \frac{p_t}{p_{T-1}} e_{T-1}^\top Ay$ , as  $\sup_{\leq T-1}(y) = \sup_{\leq t}(y)$ . A similar argument show that Eq. (7)–(8) show that all strategies in the support of  $y$  are best responses against  $x$ . ◀

When  $P_K = 1$ , then Alice has zero payoff on playing time  $K$ . Thus as long as  $K \geq 2$ , when  $P_K = 1$  there is no coinciding Nash equilibrium  $(x, y)$  where  $\text{sup}(x) = \text{sup}(y) = \{K\}$ . A similar argument applies when  $p_K = 1$ . We will therefore exclude the case  $T = K$  and either  $p_K = 1$  or  $P_K = 1$  for the next definition and Lemma 19.

► **Definition 18.** We define the values  $q_T^A, q_T^B$ , for  $T = 2, \dots, K$ , and  $r_T^A, r_T^B, z_T^A, z_T^B$ , for  $T = 1, \dots, K-1$ . The values  $z_K^A, r_K^A$  are not defined when  $T = K, p_K = 1$  and  $z_K^B, r_K^B$  are

not defined when  $T = K, P_K = 1$ .

$$\begin{aligned} q_i^A &= \frac{1}{p_i} \left( \frac{1}{P_{i-1}} - \frac{1}{P_i} \right), & q_i^B &= \frac{1}{\bar{P}_i} \left( \frac{1}{p_{i-1}} - \frac{1}{p_i} \right), \\ r_T^A &= \frac{1}{\bar{P}_T} \left( \frac{1}{P_K} - \sum_{i=T+1}^K \bar{p}_i q_i^A \right), & r_T^B &= \frac{1}{\bar{P}_T} \left( \frac{1}{p_K} - \sum_{i=T+1}^K \bar{P}_i q_i^B \right), \\ z_T^A &= r_T^A + \sum_{i=T+1}^K q_i^A, & z_T^B &= r_T^B + \sum_{i=T+1}^K q_i^B. \end{aligned}$$

► **Lemma 19.** *For every  $1 \leq T \leq K - 1$ , and the case  $T = K$  and  $P_K \neq 1$ , the system of linear equations composed of the Eq. (2), (4) and (6) of Lemma 17 has a unique solution given by*

$$y_t = \begin{cases} r_T^B / z_T^B & \text{if } t = T, \\ q_t^B / z_T^B & \text{if } T < t \leq K, \\ 0 & \text{otherwise.} \end{cases}$$

**Proof.** For convenience, we drop the normalization condition  $\sum_{t=T}^K y_t = 1$  and instead scale  $y$  such that the expected payoff of a best response is 1. That is, we replace Eq. (6) by Eq. (12):

$$e_T^T A y = 1. \quad (12)$$

Clearly the solutions of Eq. (2),(4),(6) and the solutions of Eq. (2),(4),(12) differ only by a constant multiplicative factor, and from a solution of the latter system a solution of the former one can be obtained by dividing it coordinate-wise by  $\sum_{t=T}^K y_t$ .

We first want to find the solutions of Eq. (2) and (12). Together, they can be expressed in matrix form as:

$$\begin{bmatrix} p_T \bar{P}_T & p_T & p_T & \cdots & p_T \\ p_{T+1} \bar{P}_T & p_{T+1} \bar{P}_{T+1} & p_{T+1} & \cdots & p_{T+1} \\ p_{T+2} \bar{P}_T & p_{T+2} \bar{P}_{T+1} & p_{T+2} \bar{P}_{T+2} & \cdots & p_{T+2} \\ \vdots & & & \ddots & \vdots \\ p_K \bar{P}_T & p_K \bar{P}_{T+1} & p_K \bar{P}_{T+2} & \cdots & p_K \bar{P}_K \end{bmatrix} \begin{bmatrix} y_T \\ y_{T+1} \\ y_{T+2} \\ \vdots \\ y_K \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix}. \quad (13)$$

We can simplify the above system of linear equations as follows:

$$\begin{bmatrix} 0 & P_{T+1} & 0 & \cdots & 0 \\ 0 & 0 & P_{T+2} & \cdots & 0 \\ 0 & 0 & 0 & \cdots & 0 \\ \vdots & & & \ddots & \vdots \\ \bar{P}_T & \bar{P}_{T+1} & \bar{P}_{T+2} & \cdots & \bar{P}_K \end{bmatrix} \begin{bmatrix} y_T \\ y_{T+1} \\ y_{T+2} \\ \vdots \\ y_K \end{bmatrix} = \begin{bmatrix} 1/p_T - 1/p_{T+1} \\ 1/p_{T+1} - 1/p_{T+2} \\ 1/p_{T+2} - 1/p_{T+3} \\ \vdots \\ 1/p_K \end{bmatrix}.$$

From this we can see that, for every  $1 \leq T \leq K$ , Eq. (13) has a unique solution, given by  $y_T = r_T^B$ , and  $y_t = q_t^B$ , for  $T < t \leq K$ . ◀

► **Definition 20.** Define  $T_A^*$  (respectively  $T_B^*$ ) as the smallest integer  $1 \leq T \leq K$  such that  $r_T^A > 0$  (respectively  $r_T^B > 0$ ). When  $T_A^* = T_B^*$  we denote their common value as  $T^*$ .

This definition makes sense as in the case  $p_K = 1$  (when  $r_K^A$  is not defined) we see that  $r_{K-1}^A > 0$  and otherwise  $r_K^A = \frac{1}{p_K P_K} > 0$ . A similar argument applies for  $r_K^B$ .

► **Lemma 21.** Eq. (2)–(6) have a solution if and only if  $T = T_B^*$ . In the case  $T = T_B^*$ , the solution is unique and is given by

$$y_t = \begin{cases} r_{T_B^*}^B / z_{T_B^*}^B & \text{if } t = T_B^*, \\ q_t^B / z_{T_B^*}^B & \text{if } T_B^* < t \leq K. \end{cases}$$

**Proof.** Refer [14]. ◀

The following theorem characterizes the coinciding Nash equilibria in a stingy quantum race.

► **Theorem 22.** A stingy quantum race defined by probabilities  $0 < p_1 < \dots < p_K \leq 1$  and  $0 < P_1 < \dots < P_K \leq 1$  has a coinciding Nash equilibrium if and only if  $T_A^* = T_B^*$ . In this case, letting  $T^* = T_A^* = T_B^*$  there is a unique coinciding equilibrium given by

$$x_t = \begin{cases} r_{T^*}^A / z_{T^*}^A & \text{if } t = T^*, \\ q_t^A / z_{T^*}^A & \text{if } T^* < t \leq K, \end{cases}, \quad y_t = \begin{cases} r_{T^*}^B / z_{T^*}^B & \text{if } t = T^*, \\ q_t^B / z_{T^*}^B & \text{if } T^* < t \leq K. \end{cases}$$

In particular, when  $p_i = P_i$  for all  $1 \leq i \leq K$  then  $(x, x)$  is the unique coinciding Nash equilibrium.

**Proof.** By Lemma 17 a coinciding Nash equilibrium  $(x, y)$  supported on  $\{T, T+1, \dots, K\}$  must satisfy Eq. (2)–(11). By Theorem 21, Eq. (2)–(6) are satisfied if and only if  $T = T_B^*$  and  $y$  is given as in the Lemma. We can also apply Theorem 21 to (the transpose of) Eq. (7)–(11) to see that they have a solution if and only if  $T = T_A^*$  and  $x$  is given by

$$x_t = \begin{cases} r_{T_A^*}^A / z_{T_A^*}^A & \text{if } t = T_A^*, \\ q_t^A / z_{T_A^*}^A & \text{if } T_A^* < t \leq K, \\ 0 & \text{otherwise.} \end{cases}$$

As  $x$  and  $y$  must have the same support in a coinciding Nash equilibrium, there can only exist a coinciding Nash equilibrium if  $T_A^* = T_B^*$ .

When  $p_i = P_i$  for all  $1 \leq i \leq K$  then clearly  $r_T^A = r_T^B$  and  $q_i^A = q_i^B$  and it will always be the case that  $T_A^* = T_B^*$ . Thus there will always exist a Nash equilibrium in this case, given by the unique solution to Eq. (2)–(11). ◀

### 3.2 Payoff and collision probability

In this section we will explore the consequences of the coinciding Nash equilibrium we have found for the payoff of the game and for the probability that the two players win at the same time, the collision probability. For these results, we will only consider the symmetric case when there is always a unique symmetric equilibrium whose support begins at  $T^* = T_A^* = T_B^*$ . Note that the payoff for each player with this strategy is  $\frac{1}{z_{T^*}}$ . Since a player receives payoff 1 upon winning,  $\frac{1}{z_{T^*}}$  is also exactly the each player's winning probability.

To investigate the collision probability, we will also make the following definitions.

► **Definition 23** (Unnormalized collision probability). Define

$$\sigma(T) = p_T^2 r_T^2 + \sum_{i=T+1}^K p_i^2 q_i^2.$$



With this definition,  $\frac{1}{z_{T^*}^2}\sigma(T^*)$  is the collision probability we are interested in. First we analyze the payoff in a symmetric stingy quantum race.

► **Theorem 24.** *Let  $p_1 < p_2 < \dots < p_K$  define a stingy symmetric quantum race. Then  $z_{T^*} = 1 + \sqrt{1 + \frac{1}{p_K^2} + \sigma(T^*)}$ . In particular,  $\frac{1}{z_{T^*}} \leq \sqrt{2} - 1$ .*

**Proof.** Refer [14]. ◀

► **Corollary 25.** *If  $T^* \geq 2$  then*

$$p_{T^*-1} \leq \sqrt{2} - 1 .$$

**Proof.** As can be seen from Bob playing time  $T^* - 1$ , we have  $p_{T^*-1}z_{T^*} \leq 1$ , thus  $p_{T^*-1} \leq \sqrt{2} - 1$  by Theorem 24. ◀

Although Theorem 24 gives an exact expression for the payoff, we would like to get a general lower bound on the payoff. This requires showing an upper bound on the collision probability. Showing an upper bound on the collision probability is also important for the application to Bitcoin, to estimate the forking probability amongst quantum miners.

The first step to upper bounding the collision probability is to get a rough lower bound on  $p_{T^*}$ . This is our initial bootstrap, which will then let us upper bound the collision probability and then in turn get a sharper lower bound on  $p_{T^*}$  in Corollary 28. For these results we restrict to  $\ell$ -dense stingy quantum races.

► **Lemma 26.** *Let  $p_1, \dots, p_K$  define an  $\ell$ -dense symmetric stingy quantum race. If  $K \geq 6\ell$  then  $p_{T^*} > \frac{5}{21}$ . In particular,  $T^* \geq 2$ .*

**Proof.** Refer [14]. ◀

► **Theorem 27.** *Let  $p_1, \dots, p_K$  define an  $\ell$ -dense symmetric stingy quantum race. If  $K \geq 6\ell$  then*

$$\frac{\sigma(T^*)}{z_{T^*}^2} \leq \frac{6\ell}{K} \quad \text{and} \quad \sigma(T^*) \leq \frac{196\ell}{K} .$$

**Proof.** Refer [14]. ◀

Now that we have an upper bound on the collision probability, we obtain the following corollary to Theorem 24.

► **Corollary 28.** *Let  $p_1, \dots, p_K$  define an  $\ell$ -dense symmetric stingy quantum race. Let  $\tau = \frac{50\sqrt{2}\ell}{K}$ . If  $K \geq 6\ell$  then*

$$z_{T^*} \leq \sqrt{2} + 1 + \tau, \quad \frac{1}{z_{T^*}} \geq \sqrt{2} - 1 - \tau(\sqrt{2} - 1)^2, \quad p_{T^*} \geq \sqrt{2} - 1 - \tau(\sqrt{2} - 1)^2 .$$

**Proof.** Refer [14]. ◀

## 4 Two-player quantum races

In this section, we bootstrap our results about symmetric stingy quantum races to analyze symmetric quantum races. Our main results are two-fold.

1. The unique coinciding Nash equilibrium in an  $\ell$ -dense symmetric stingy quantum race is an approximate Nash equilibrium in the corresponding quantum race.
2. The approximate Nash equilibrium from (1) achieves a payoff which is nearly optimal among all symmetric Nash equilibria in a symmetric quantum race.

The intuition for item (1) is clear. The only difference between a stingy quantum race and a quantum race is the payoff on ties. For the unique coinciding Nash equilibrium we have shown that the collision probability is  $O(\ell/K)$ , thus the change in payoff on ties will make only a small change to the payoffs under this strategy.

For item (2), we use the quadratic programming characterization of Nash equilibria [15]. Consider a game  $(A, B)$  where  $A, B$  are  $m$ -by- $n$  matrices. The program

$$\begin{aligned} & \underset{x \in \Delta^m, y \in \Delta^n, \alpha, \beta \in \mathbb{R}}{\text{maximize}} && x^T(A + B)y - \alpha - \beta \\ & \text{subject to} && Ay \leq \alpha \mathbf{1}, \\ & && B^T x \leq \beta \mathbf{1}, \end{aligned} \tag{14}$$

has an optimal value of 0, and any  $(x, y)$  attaining the value 0 is a Nash equilibrium. In the case of a symmetric quantum race  $(A, A^T)$ , when we restrict to symmetric strategies  $(x, x)$  the objective function in Eq. (14) becomes negative definite plus linear, making this a standard quadratic program. We then use the tight dual formulation of a quadratic program [8] to give an upper bound on the payoff of any symmetric Nash equilibrium, by explicitly constructing solutions to the dual problem. This allows us to show that the payoff of the unique coinciding equilibrium in a stingy race achieves a payoff which is within  $O(\sqrt{\ell/K})$  of optimal amongst all symmetric equilibria in the corresponding quantum race.

We now proceed to show these two results.

### 4.1 Approximate Nash equilibrium

► **Definition 29.** A two-player game described by payoff matrices  $(A, B)$  is said to have an  $\epsilon$ -approximate Nash equilibrium  $(p, q)$ , for  $\epsilon \geq 0$ , if the following two conditions hold

$$p^T Aq \geq v^T Aq - \epsilon \text{ for all } v \in \Delta_m \tag{15}$$

$$p^T Bq \geq p^T Bu - \epsilon \text{ for all } u \in \Delta_n . \tag{16}$$

► **Definition 30.** A two-player game described by payoff matrices  $(A, B)$  is said to have an  $\epsilon$ -well supported Nash equilibrium  $(p, q)$ , for  $\epsilon \geq 0$  if

$$e_i^T Aq \geq e_j^T Aq - \epsilon \text{ for all } i \in \text{sup}(p) \text{ and } j \in [m]$$

$$p^T B e_i \geq p^T B e_j - \epsilon \text{ for all } i \in \text{sup}(q) \text{ and } j \in [n] .$$

Note that an  $\epsilon$ -well supported Nash equilibrium is also an  $\epsilon$ -approximate Nash equilibrium, but the reverse does not hold.

► **Theorem 31.** Let  $p_1, \dots, p_K$  be an  $\ell$ -dense sequence defining the symmetric stingy quantum race  $(A_0, A_0^T)$  and the symmetric quantum race  $(A, A^T)$ . Let  $(x, x)$  be the unique coinciding Nash equilibrium for the stingy quantum race  $(A_0, A_0^T)$  given by Theorem 22. Then  $(x, x)$  is a  $\frac{7(\sqrt{2}-1)\ell}{K}$ -well supported Nash equilibrium in the quantum race  $(A, A^T)$ .

**Proof.** To show that  $(x, x)$  is an  $\epsilon$ -well supported Nash equilibrium in the quantum race  $(A, A^T)$  it suffices to show that  $e_i^T Ax \geq e_j^T Ax - \epsilon$  for all  $T^* \leq i \leq K$  and  $j \in [K]$ . We omit the details of the proof here and refer the readers to [14]. ◀

## 4.2 Upper bound on payoff

Let  $(x, x)$  be the unique coinciding Nash equilibrium in an  $\ell$ -dense symmetric stingy quantum race  $(A_0, A_0^T)$ . We have just shown that  $(x, x)$  is a  $\frac{7(\sqrt{2}-1)\ell}{K}$ -well supported Nash equilibrium in the corresponding quantum race  $(A, A^T)$ . By Corollary 28,  $(x, x)$  achieves payoff at least  $\sqrt{2} - 1 - 50\sqrt{2}(\sqrt{2} - 1)^2 \frac{\ell}{K}$  in the game  $(A, A^T)$ . In this section, we show that this payoff is within  $O(\sqrt{\ell/K})$  of optimal among all symmetric strategies  $(y, y)$  for the game  $(A, A^T)$ .

Our starting point is to use the program in Eq. (14) to provide a means to upper bound the value of any symmetric equilibrium.

► **Lemma 32.** *Let  $(A, A^T)$  be a symmetric game and define for  $c \geq 0$*

$$\begin{aligned} \gamma_A(c) = \underset{x}{\text{maximize}} \quad & \frac{1}{2}x^T(A + A^T)x \\ \text{subject to} \quad & Ax \leq c\mathbf{1}, \\ & \mathbf{1}^T x = 1, x \geq \mathbf{0}. \end{aligned}$$

*For all  $c_0$ , such that  $\gamma_A(c) < c$  for all  $c \geq c_0$ , the payoff of any symmetric Nash equilibrium in the game  $(A, A^T)$  is less than  $c_0$ .*

**Proof.** We show the contrapositive. Suppose there is a symmetric Nash equilibrium  $(x, x)$  with payoff  $c \geq c_0$ . Then  $(x, x, c, c)$  is a feasible solution to the program in Eq. (14) with objective value 0. Thus  $Ax \leq c$  and  $\frac{1}{2}x^T(A + A^T)x = c$ . ◀

This is the approach we take to upper bounding the payoff of symmetric Nash equilibria in a quantum race.

► **Theorem 33.** *Let  $p_1, \dots, p_K$  be an  $\ell$ -dense sequence with  $K \geq 6\ell$ . Then any symmetric Nash equilibrium  $(x, x)$  in the two-player quantum race defined by  $p_1, \dots, p_K$  has payoff at most  $\sqrt{2} - 1 + 5\sqrt{\frac{\ell}{K}}$ .*

**Proof.** Let  $(A, A^T)$  be the payoff matrices of a two-player quantum race defined by  $p_1, \dots, p_K$ . We will show that  $\gamma_A(c) < c$  for all  $c > \sqrt{2} - 1 + 5\sqrt{\frac{\ell}{K}}$ . By Lemma 32 this proves the theorem.

In the case of a quantum race  $A + A^T = p\mathbf{1}^T + 1p^T - pp^T$ . This means that over the probability simplex, the quadratic form  $x^T(A + A^T)x = -x^T(pp^T)x + 2p^T x$  is a negative-definite plus linear function. In this case,  $\gamma_A(c)$  is in the standard form of a quadratic program and has a dual program with matching value [8].

$$\begin{aligned} \gamma_A(c) = \underset{v \in \mathbb{R}^K, \lambda, d \in \mathbb{R}}{\text{minimize}} \quad & \frac{1}{2}\lambda^2 + c \cdot \mathbf{1}^T v + d \\ \text{subject to} \quad & A^T v \geq (1 - \lambda)p - d\mathbf{1}, \\ & v \geq \mathbf{0}. \end{aligned} \tag{17}$$

Our approach will be to construct a feasible solution to Eq. (17) to demonstrate that  $\gamma_A(c) < c$  for all  $c > \sqrt{2} - 1 + 5\sqrt{\frac{\ell}{K}}$ .

## 51:16 Strategies for Quantum Races

First note that for  $c > \frac{1}{2}$  there is a trivial solution where  $\lambda = 1, d = 0$  and  $v$  is the all-zero vector which shows that  $\gamma(c) < c$ . We now focus on the case  $c \leq \frac{1}{2}$ . Let  $\sqrt{2} - 1 \leq c \leq \frac{1}{2}$ . We will develop a lower bound on  $c$  which implies  $\gamma_A(c) < c$ .

Let  $S$  be the smallest index  $i$  such that  $p_i \geq c$ . Note that as  $A$  is an  $\ell$ -dense quantum race we have  $p_S \leq c + \frac{\ell}{K}$ . We let

$$d = (1 - \lambda) \left( 1 + p_K - \frac{p_K}{p_S} \right)$$

and

$$v(i) = \begin{cases} 0 & \text{if } 1 \leq i < S \\ (1 - \lambda - d) \frac{p_S}{\bar{p}_S} \frac{1}{p_i} \left( \frac{1}{p_i} - \frac{1}{p_{i+1}} \right) & \text{if } S \leq i < K \\ (1 - \lambda - d) \frac{p_S}{p_K^2 \bar{p}_S} - \frac{(1-\lambda)}{p_K} & \text{if } i = K \end{cases} .$$

The choice of  $v$  comes from solving the system of linear equations  $(A_0^T v)_i = (1 - \lambda)p_i - d$  for  $S \leq i \leq K$ . The parameter  $\lambda$  will be chosen later.

Let us see that  $v$  satisfies the constraints of Eq. (17). Note that  $1 - \lambda - d = (1 - \lambda)p_K \frac{\bar{p}_S}{p_S}$ . Thus  $v(K) = 0$  and  $v \geq 0$  so long as  $\lambda \leq 1$ .

As mentioned, by construction  $v$  satisfies  $(A_0^T v)_i = (1 - \lambda)p_i - d$  for  $S \leq i \leq K$ . Thus as  $A = A_0 + \frac{1}{2} \text{diag}(p)^2$  and  $v \geq 0$  we also have  $(A^T v)_i \geq (1 - \lambda)p_i - d$  for  $S \leq i \leq K$ .

For  $i < S$  we have that

$$(A^T v)_i \geq (A_0^T v)_i = \bar{p}_i p^T v \geq \bar{p}_S p^T v = (1 - \lambda)p_S - d \geq (1 - \lambda)p_i - d .$$

Thus the constraint  $A_0^T v \geq (1 - \lambda)p - d\mathbf{1}$  is satisfied. We have shown that  $v$  is a feasible solution for any choice of  $\lambda \leq 1$ . We now choose  $\lambda$  to minimize the objective value. Substituting our choices of  $v, d$  into the objective value we have

$$\begin{aligned} \gamma_A(c) &\leq \frac{1}{2} \lambda^2 + (1 - \lambda) \left( 1 + p_K - \frac{p_K}{p_S} \right) + c(1 - \lambda) p_K \sum_{i=S}^{K-1} \frac{1}{p_i} \left( \frac{1}{p_i} - \frac{1}{p_{i+1}} \right) \\ &= \frac{1}{2} \lambda^2 + (1 - \lambda) \left( 1 + p_K \left( -\frac{\bar{p}_S}{p_S} + c \cdot \sum_{i=S}^{K-1} \frac{1}{p_i} \left( \frac{1}{p_i} - \frac{1}{p_{i+1}} \right) \right) \right) \end{aligned}$$

Define

$$\beta(c) = 1 + p_K \left( -\frac{\bar{p}_S}{p_S} + c \cdot \sum_{i=S}^{K-1} \frac{1}{p_i} \left( \frac{1}{p_i} - \frac{1}{p_{i+1}} \right) \right) .$$

The objective value  $\frac{1}{2} \lambda^2 + (1 - \lambda)\beta(c)$  is minimized over  $\lambda$  by taking  $\lambda = \beta(c)$ . This makes the objective value  $\beta(c) - \beta(c)^2/2$ . We have now reduced the problem to showing  $\beta(c) - \beta(c)^2/2 - c < 0$ . The roots of the corresponding quadratic equation are  $1 \pm \sqrt{1 - 2c}$ . Note that  $c \leq 1/2$ , thus the square root term will be real. Thus we will simultaneously have  $\beta(c) \leq 1$  and  $\beta(c) - \beta(c)^2/2 < c$  when  $\beta(c) < 1 - \sqrt{1 - 2c}$ . In Lemma 34, we show that  $\beta(c) < 1 - \sqrt{1 - 2c}$  when  $\sqrt{2} - 1 + 5\sqrt{\frac{\ell}{K}} \leq c \leq \frac{1}{2}$ . This will conclude the proof.  $\blacktriangleleft$

**► Lemma 34.**  $\beta(c) < 1 - \sqrt{1 - 2c}$  for any  $\sqrt{2} - 1 + 5\sqrt{\frac{\ell}{K}} \leq c \leq \frac{1}{2}$ .

**Proof.** Refer [14].  $\blacktriangleleft$

## 5 Multiplayer quantum races

### 5.1 Basic properties

For an integer  $n \geq 2$ , an  $n$ -player game is specified by a set of *pure strategies*  $S_i$ , and *payoff* functions  $u_i : S \rightarrow \mathbb{R}$ , for each player  $i \in [n]$ , where by definition  $S = S_1 \times \cdots \times S_n$  is the set of *pure strategy profiles*. For  $s \in S$ , the value  $u_i(s)$  is the payoff of player  $i$  for pure strategy profile  $s$ . Let  $S_{-i} = S_1 \times \cdots \times S_{i-1} \times S_{i+1} \times \cdots \times S_n$  be the set of all pure strategy profiles of players other than  $i$ . For  $s \in S$  and  $i \in [n]$ , we set the *partial* pure strategy profile  $s_{-i} \in S_{-i}$  to be  $(s_1, \dots, s_{i-1}, s_{i+1}, \dots, s_n)$ . For  $s'$  in  $S_{-i}$ , and  $s_i \in S_i$ , we denote by  $(s', s_i)$  the *combined* pure strategy profile  $(s'_1, \dots, s'_{i-1}, s_i, s'_{i+1}, \dots, s'_n) \in S$ . We will suppose that each player has  $m$  pure strategies and that  $S_i = \{e_1, \dots, e_m\}$ , the canonical basis of the vector space  $\mathbb{R}^m$ , for all  $i \in [n]$ , and therefore  $S = \{e_1, \dots, e_m\}^n$ . For simplicity, instead of  $e_j$  we often say strategy  $j$ .

A *mixed strategy* for player  $i$  is a probability distribution over  $S_i$  that we identify with a vector  $x_i = (x_i^1, \dots, x_i^m)$  such that  $x_i^j \geq 0$ , for all  $j \in [m]$ , and  $\sum_{j \in [m]} x_i^j = 1$ . We denote by  $\Delta_i$  the set of mixed strategies for  $i$ , and we call  $\Delta = \Delta_1 \times \cdots \times \Delta_n$  the set of *mixed strategy profiles*. For a mixed strategy profile  $x = (x_1, \dots, x_n)$  and pure strategy profile  $s \in S$ , the product  $x^s = x_1^{s_1} x_2^{s_2} \cdots x_n^{s_n}$  is the probability of  $s$  in  $x$ . We will consider the multilinear extension of the payoff functions from  $S$  to  $\Delta$  defined by  $u_i(x) = \sum_{s \in S} x^s u_i(s)$ . The set  $\Delta_{-i}$ , the partial mixed strategy profile  $x_{-i}$ , for  $x \in \Delta$  and  $i \in [n]$ , and the combined mixed strategy profile  $(x', x_i)$  for  $x' \in \Delta_{-i}$  and  $x_i \in \Delta_i$  are defined analogously to the pure case.

The pure strategy  $s_i$  is a *best response* for player  $i$  against the partial mixed strategy profile  $x' \in \Delta_{-i}$  if it maximizes  $u_i(x', \cdot)$ . For  $x \in \Delta$  and  $i \in [n]$ , we will denote by  $\text{br}(x_{-i})$  the set of best responses of player  $i$  against  $x_{-i}$ . A *Nash equilibrium* is a mixed strategy profile  $x = (x_1, \dots, x_n)$  such that  $\text{sup}(x_i) \subseteq \text{br}(x_{-i})$ , for all  $i \in [n]$ .

► **Definition 35** ( $n$ -party stingy quantum race). Let  $n \geq 2$  be a positive integer. An  $n$ -party stingy quantum race is defined by a sequence of increasing probabilities  $0 < P_1 < P_2 < \dots < P_K \leq 1$ , for some positive integer  $K$ . The set of pure strategies of all players is  $[K]$ . For every  $i$ , the utility function of the  $i^{\text{th}}$  player is defined as

$$u_i(s_1, \dots, s_n) = P_{s_i} \prod_{k \neq i, s_k \leq s_i} \bar{P}_{s_k}.$$

Consider an  $n$ -party stingy quantum race given by the probabilities  $0 < P_1 < \dots < P_K \leq 1$ , and let  $x_{-i} \in \Delta_{-i}$  for some  $i \in [n]$ . If player  $i$  plays the pure strategy  $s$  against  $x_{-i}$ , her payoff is

$$u_i(x_{-i}, s) = P_s \prod_{k \neq i} \left( \sum_{s_k \in \text{sup}_{\leq s}(x_k)} x_k^{s_k} \bar{P}_{s_k} + \sum_{s_k \in \text{sup}_{> s}(x_k)} x_k^{s_k} \right). \quad (18)$$

The following is the multiparty analog of Claim 14.

► **Claim 36.** Let  $x = (x_1, \dots, x_n)$  be a Nash equilibrium of an  $n$ -party stingy quantum race defined by probabilities  $P_1 < \dots < P_K$ . If  $s_1 \in \text{sup}(x_i)$ , for some  $i \in [n]$ , then for all  $s_2 > s_1$  there exists  $k \neq i$  such that  $\text{sup}_{\leq s_1}(x_k) \neq \text{sup}_{\leq s_2}(x_k)$ .

**Proof.** If  $\text{sup}_{\leq s_1}(x_k) = \text{sup}_{\leq s_2}(x_k)$ , for all  $k \neq i$  then

$$u_i(x_{-i}, s_2) = \frac{P_{s_2}}{P_{s_1}} u_i(x_{-i}, s_1).$$

As  $p_{s_1} < p_{s_2}$ , the payoff for playing  $s_2$  is strictly larger than that for playing  $s_1$ . Therefore  $s_1$  is not a best response for  $x_{-i}$ , in contradiction with the definition of a Nash equilibrium. ◀

This claim implies the following properties for the supports of Nash equilibria in a multiplayer stingy quantum race.

► **Corollary 37.** *Let  $x$  be a Nash equilibrium of an  $n$ -party stingy quantum race defined by probabilities  $0 < P_1 < P_2 < \dots < P_K \leq 1$ . Then we have:*

- $\bigcup_{i=1}^n \text{sup}(x_i)$  is an interval containing  $K$ ,
- for every  $i \in [n]$ , for every  $s_1, s_2 \in \text{sup}(x_i)$  there exists  $s \in \bigcup_{k \neq i} \text{sup}(x_k)$  with  $s_1 < s \leq s_2$ .

Let  $x$  be a Nash equilibrium of an  $n$ -party stingy quantum race. We say that  $x$  is *coinciding* if  $\text{sup}(x_i) = \text{sup}(x_k)$ , for all  $i, k \in [n]$ . We call this common support in a coinciding Nash equilibrium the *support* of the equilibrium. In the multiparty case we will only study coinciding Nash equilibria.

## 5.2 Coinciding Nash equilibria of stingy multiplayer races

By Corollary 37 we know that in a coinciding Nash equilibrium of an  $n$ -party stingy quantum race the support of the equilibrium is of the form  $\{T, T+1, \dots, K\}$ , for some  $1 \leq T \leq K$ . We would like to characterize these coinciding equilibria.

► **Lemma 38.** *Let  $x = (x_1, \dots, x_n)$ , where  $x_i$  is a  $K$ -dimensional real vector for every  $i \in [n]$ , and let  $1 \leq T \leq K$ . Then  $x$  is a Nash equilibrium of support  $\{T, T+1, \dots, K\}$  in an  $n$ -party stingy quantum race defined by  $0 < P_1 < \dots < P_K \leq 1$  if and only if  $x$  satisfies the following system, for all  $i \in [n]$ :*

$$u_i(x_{-i}, t) = u_i(x_{-i}, T) \quad \text{for } T < t \leq K, \quad (19)$$

$$u_i(x_{-i}, T-1) \leq u_i(x_{-i}, T), \quad (20)$$

$$x_i^t = 0 \quad \text{for } 0 < t < T, \quad (21)$$

$$x_i^t > 0 \quad \text{for } T \leq t \leq K, \quad (22)$$

$$\sum_{t=T}^K x_i^t = 1. \quad (23)$$

**Proof.** For every  $i \in [n]$ , Eq. (21)–(23) express that  $x_i$  is a probability distribution of support  $\{T, T+1, \dots, K\}$ . For  $T \geq 2$ , when playing a strategy  $t < T$  against the partial mixed strategy profile  $x_{-i}$ , the  $i$ th player's payoff is maximized if she plays  $T-1$ . Therefore Eq. (19) and (20) express that the strategies in her support are all best responses against  $x_{-i}$ . ◀

► **Definition 39.** For an  $n$ -party stingy quantum race defined by  $0 < P_1 < \dots < P_K \leq 1$  we define its *reduced game* as the 2-party stingy quantum race defined by the two sequences of probabilities  $p_1 < \dots < p_K$ , and  $P_1 < \dots < P_K$  where  $p_j = P_j^{1/(n-1)}$ , for  $1 \leq j \leq K$ .

We denote by  $A$  the payoff matrix of the first player in the reduced game

► **Lemma 40.** *Let an  $n$ -party stingy quantum race be defined by  $0 < P_1 < \dots < P_K \leq 1$ , let  $x = (x_1, \dots, x_n)$ , where  $x_i$  is a  $K$ -dimensional vector, and let  $1 \leq T \leq K$ . Then Eq. (19)–(23) are satisfied by  $x$ , for every  $i \in [n]$  if and only if Eq. (2)–(6) for the reduced game are satisfied by  $x_i$ , for every  $i \in [n]$ .*

**Proof.** Refer [14]. ◀

The following theorem characterizes the coinciding Nash equilibria in an  $n$ -party stingy quantum race.

► **Theorem 41.** *An  $n$ -party stingy quantum race always has a unique coinciding Nash equilibrium  $x = (x_1, \dots, x_n)$ , where  $x_1 = \dots = x_n$ . If the game is defined by the probabilities  $0 < P_1 < P_2 < \dots < P_K$  then the coinciding equilibrium has support  $\{T^*, T^* + 1, \dots, K\}$ , where  $T^* = T_B^*$  of the reduced game, and for all  $i \in [n]$ , the distribution  $x_i$  is defined on its support as*

$$x_i^t = \begin{cases} r_{T^*}^B / z_{T^*}^B & \text{if } t = T^*, \\ q_t^B / z_{T^*}^B & \text{if } T^* < t \leq K. \end{cases}$$

**Proof.** Combining Lemma 38 and Lemma 40, we get that  $x$  is a coinciding Nash equilibrium of support  $\{T, T + 1, \dots, K\}$  if and only if  $x_i$  satisfies Eq. (2)–(6) for the reduced game, for all  $i \in [n]$ . By Theorem 21 this happens if and only if  $T = T_B^*$  of the reduced game, and the unique solution for  $x_i$ , for  $i \in [n]$ , is the one stated by the Theorem. ◀

### 5.3 Collision probability of the stingy coinciding equilibrium

Our main objective in this section is to upper bound the collision probability – the probability that two or more players succeed at the same time – in the coinciding equilibrium found in the last section for an  $\ell$ -dense stingy  $n$ -player quantum race. To help with this, we make the following definition.

► **Definition 42.** For a joint probability distribution  $y = (y_1, \dots, y_n) \in \Delta$ , let  $\text{cp}_i^m(y)$  denote the probability that player  $i$  succeeds first and that exactly  $m$  players (including  $i$ ) succeed at the same time under the joint strategy  $y$ . Let  $\text{cp}_i(y) = \sum_{m=2}^n \text{cp}_i^m(y)$  denote the probability that player  $i$  succeeds first and at least one other player succeeds at the same time.

Let us also set up notation to describe the coinciding equilibrium in a stingy multiplayer race. Define the following quantities

$$q_i = \frac{1}{\bar{P}_i} \left( \frac{1}{P_{i-1}^{1/(n-1)}} - \frac{1}{P_i^{1/(n-1)}} \right), r_T = \frac{1}{\bar{P}_T} \left( \frac{1}{P_K^{1/(n-1)}} - \sum_{i=T+1}^K \bar{P}_i q_i \right), z_T = r_T + \sum_{i=T+1}^K q_i .$$

Let  $T^*$  be the starting point of the support of the coinciding equilibrium. Then by Theorem 41, the strategy of player  $i$  in the coinciding equilibrium is given by

$$x_i^t = \begin{cases} r_{T^*} / z_{T^*} & \text{if } t = T^*, \\ q_t / z_{T^*} & \text{if } T^* < t \leq K, \\ 0 & \text{if } t < T^* . \end{cases} \quad (24)$$

To obtain concrete bounds on the collision probability, we will need bounds on  $z_{T^*}$  and  $P_{T^*-1}$ .

► **Lemma 43.** *In any stingy multiplayer quantum race with  $n$  players  $(1/z_{T^*})^{n-1} < 1/n$ .*

**Proof.** Refer [14]. ◀

► **Theorem 44.** *Let  $P_1, \dots, P_K$  define a stingy  $n$ -player quantum race with  $n \geq 2$ . Then  $P_{T^*-1} < \frac{1}{n}$ . If in addition  $P_1, \dots, P_K$  form an  $\ell$ -dense sequence and  $4e\ell n \leq K$  then  $P_{T^*-1} \geq \frac{1}{2en}$ , where  $e$  is Euler's number.*

**Proof.** Refer [14]. ◀



The next lemma bounds the collision probability for player  $i$  when all players but player  $i$  play according to the coinciding equilibrium, and player  $i$  plays an arbitrary strategy  $v$ . We will use this lemma to bound the total collision probability and also in Section 5.4 to show that the stingy coinciding equilibrium is an approximate equilibrium in a multiparty race.

► **Lemma 45.** *Let  $P_1, \dots, P_K$  define an  $\ell$ -dense stingy  $n$ -player quantum race,  $n \geq 2$ , with  $4en\ell \leq K$ , and let  $x$  be the unique coinciding equilibrium given by Eq. (24). Then  $\text{cp}_i(x_{-i}, v) \leq \frac{8e\ell}{K}$  for any  $i \in [n]$  and  $v \in \Delta_i$ .*

**Proof.** Refer [14]. ◀

► **Theorem 8.** *Let  $P_1, \dots, P_K$  define an  $\ell$ -dense stingy  $n$ -player quantum race such that  $4en\ell \leq K$ . When the players play the coinciding equilibrium of the stingy race, the probability that two or more players succeed at the same time is at most  $\frac{8en\ell}{K}$ .*

**Proof.** By Lemma 45, the probability that two or more players succeed at the same time is at most

$$\sum_{i=1}^n \text{cp}_i(x) \leq \frac{8en\ell}{K} . \quad \blacktriangleleft$$

## 5.4 Multiplayer quantum races

In this section we use our results about the stingy multiplayer quantum race to analyse the multiplayer quantum race. Namely, we show that the coinciding equilibrium in a stingy multiplayer quantum race is an approximate equilibrium in a multiplayer race. The difference between a stingy multiplayer race and a multiplayer race is that in a multiplayer race, the payoff is equally divided amongst all players who succeed first at the same time.

► **Definition 46** (Multiplayer quantum race). Let  $u_i$  be the payoff function for player  $i \in [n]$  in the  $n$ -player stingy race defined by  $P_1 < \dots < P_k$ , as given by Eq. (18). The payoff function  $u'_i$  in the  $n$ -player quantum race defined by  $P_1, \dots, P_k$  is

$$u'_i(x) = u_i(x) + \sum_{m=2}^n \frac{\text{cp}_i^m(x)}{m} .$$

While the tie-splitting payoff in Definition 46 is quite natural, one could imagine other definitions in-between stingy multiplayer races and the definition of multiplayer races we have given. Our results in this section depend very weakly on the exact definition of how ties are split in a multiplayer race. In fact, the only property we use is  $u'_i(x) \leq u_i(x) + \text{cp}_i(x)$ . This property holds under any reasonable definition of tie-splitting.

Now we show Theorem 9 from the introduction that the coinciding Nash equilibrium in a multiplayer stingy quantum race is an approximate Nash equilibrium in a multiplayer quantum race.

► **Theorem 9.** *Let  $P_1, \dots, P_K$  define an  $\ell$ -dense stingy  $n$ -player quantum race,  $n \geq 2$ , with  $4en\ell \leq K$ . If  $x = (x_1, \dots, x_n)$  is the coinciding Nash equilibrium for this stingy race, then  $x$  is an  $\frac{8e\ell}{K}$ -approximate Nash equilibrium of the corresponding quantum race.*

**Proof.** Refer [14]. ◀

---

**References**

---

- 1 Divesh Aggarwal, Gavin Brennen, Troy Lee, Miklos Santha, and Marco Tomamichel. Quantum attacks on Bitcoin and how to prevent against them. Technical report, arXiv, 2017. [arXiv:1710.10377](https://arxiv.org/abs/1710.10377).
- 2 Adam Back. Hashcash – a denial of service counter-measure, 2002. Available at: <http://www.hashcash.org/papers/hashcash.pdf>.
- 3 Alex Biryukov and Dmitry Khovratovich. Equihash: Asymmetric proof-of-work based on the generalized birthday problem. *Ledger*, 2:1–30, 2017.
- 4 Bitmain. Bitmain Antminer S9. [https://shop.bitmain.com/antminer\\_s9\\_asic\\_bitcoin\\_miner.html](https://shop.bitmain.com/antminer_s9_asic_bitcoin_miner.html), 2018. Accessed 2018-02-16.
- 5 Vitalik Buterin. Bitcoin is not quantum safe, and how we can fix it when needed. <https://bitcoinmagazine.com/articles/bitcoin-is-not-quantum-safe-and-how-we-can-fix-1375242150/>, 2013.
- 6 Miguel Castro and Barbara Liskov. Practical Byzantine Fault Tolerance. In *Third Symposium on Operating Systems Design and Implementation*, 1999.
- 7 Catalin Dohotaru and Peter Høyer. Exact quantum lower bound for Grover’s problem. Technical report, arXiv, 2008. [arXiv:0810.3647](https://arxiv.org/abs/0810.3647).
- 8 William S. Dorn. Duality in quadratic programming. *Quarterly of applied mathematics*, 18(2):155–162, 1960.
- 9 Ittay Eyal and Emin Gün Sirer. Majority is not enough: Bitcoin mining is vulnerable. In *18th International Conference on Financial Cryptography and Data Security*, 2014.
- 10 Arthur Gervais, Ghassan Karame, Karl Wüst, Vasileios Glykantzis, Hubert Ritzdorf, and Srdjan Capkun. On the security and performance of proof of work blockchains. In *Proceedings of the 2016 ACM SIGSAC Conference on Compute and Communications Security (CCS’16)*, pages 3–16, 2016.
- 11 Lov K. Grover. A Fast Quantum Mechanical Algorithm for Database Search. In *Proceedings of the Twenty-eighth Annual ACM Symposium on Theory of Computing*, STOC ’96, pages 212–219, New York, NY, USA, 1996. ACM. doi:10.1145/237814.237866.
- 12 Nicole Immorlica, Adam Tauman Kalai, Brendan Lucier, Ankur Moitra, Andrew Postlewaite, and Moshe Tennenholtz. Dueling algorithms. In *Proceedings of the forty-third annual ACM symposium on theory of computing (STOC’11)*, pages 215–224, 2011.
- 13 Aggelos Kiayias, Alexander Russell, Bernardo David, and Roman Oliynykov. Ouroboros: A provably secure proof-of-stake blockchain protocol. In *CRYPTO*, pages 357–388, 2017.
- 14 T. Lee, M. Ray, and M. Santha. Strategies for quantum races. *ArXiv e-prints*, September 2018. [arXiv:1809.03671](https://arxiv.org/abs/1809.03671).
- 15 Olvi L. Mangasarian and H. Stone. Two-person nonzero-sum games and quadratic programming. *Journal of mathematical analysis and applications*, 9:348–355, 1964.
- 16 Satoshi Nakamoto. Bitcoin: A peer-to-peer electronic cash system, 2009. Available at: <http://www.bitcoin.org/pdf>.
- 17 John F. Nash. Non-cooperative Games. *Annals of Mathematics*, 54(2):286–295, 1951.
- 18 Or Sattath. On the insecurity of quantum Bitcoin mining. Technical report, arXiv, 2018. Appeared in QCRYPT 2018. [arXiv:1804.08118](https://arxiv.org/abs/1804.08118).



# Lower Bounds for Tolerant Junta and Unateness Testing via Rejection Sampling of Graphs

Amit Levi<sup>1</sup>

University of Waterloo, Canada  
amit.levi@uwaterloo.ca

Erik Waingarten<sup>2</sup>

Columbia University, USA  
eaw@cs.columbia.edu

---

## Abstract

We introduce a new model for testing graph properties which we call the *rejection sampling model*. We show that testing bipartiteness of  $n$ -nodes graphs using rejection sampling queries requires complexity  $\tilde{\Omega}(n^2)$ . Via reductions from the rejection sampling model, we give three new lower bounds for tolerant testing of Boolean functions of the form  $f: \{0, 1\}^n \rightarrow \{0, 1\}$ :

- Tolerant  $k$ -junta testing with *non-adaptive* queries requires  $\tilde{\Omega}(k^2)$  queries.
- Tolerant unateness testing requires  $\tilde{\Omega}(n)$  queries.
- Tolerant unateness testing with *non-adaptive* queries requires  $\tilde{\Omega}(n^{3/2})$  queries.

Given the  $\tilde{O}(k^{3/2})$ -query non-adaptive junta tester of Blais [7], we conclude that non-adaptive tolerant junta testing requires more queries than non-tolerant junta testing. In addition, given the  $\tilde{O}(n^{3/4})$ -query unateness tester of Chen, Waingarten, and Xie [19] and the  $\tilde{O}(n)$ -query non-adaptive unateness tester of Baleshzar, Chakrabarty, Pallavoor, Raskhodnikova, and Seshadhri [3], we conclude that tolerant unateness testing requires more queries than non-tolerant unateness testing, in both adaptive and non-adaptive settings. These lower bounds provide the first separation between tolerant and non-tolerant testing for a natural property of Boolean functions.

**2012 ACM Subject Classification** Theory of computation → Probabilistic computation

**Keywords and phrases** Property Testing, Juntas, Tolerant Testing, Boolean functions

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2019.52

**Related Version** A full version of the paper is available at <https://arxiv.org/abs/1805.01074v1>.

## 1 Introduction

Over the past decades, property testing has emerged as an important line of research in sublinear time algorithms. The goal is to understand randomized algorithms for approximate decision making, where the algorithm needs to decide (with high probability) whether a huge object has some property by making a few queries to the object. Many different types of objects and properties have been studied from this property testing perspective (see the surveys by Ron [35, 36] and the recent textbook by Goldreich [26] for overviews of contemporary property testing research). This paper deals with property testing of Boolean functions  $f: \{0, 1\}^n \rightarrow \{0, 1\}$  and property testing of graphs with vertex set  $[n]$ .

---

<sup>1</sup> Research supported by NSERC Discovery grant and the David R. Cheriton Graduate Scholarship. Part of this work was done while the author was visiting Columbia University.

<sup>2</sup> This work is supported in part by the NSF Graduate Research Fellowship under Grant No. DGE-16-44869, CCF-1703925, CCF-1563155, and CCF-1420349.



In this paper we describe a new model of graph property testing, which we call the *rejection sampling model*. For  $n \in \mathbb{N}$  and a subset  $\mathcal{P}$  of graphs on the vertex set  $[n]$ , we say a graph  $G$  on vertex set  $[n]$  has property  $\mathcal{P}$  if  $G \in \mathcal{P}$  and say  $G$  is  $\varepsilon$ -far from having property  $\mathcal{P}$  if all graphs  $H \in \mathcal{P}$  differ on at least  $\varepsilon n^2$  edges<sup>3</sup>. The problem of  $\varepsilon$ -testing  $\mathcal{P}$  with *rejection sampling queries* is the following task:

Given some  $\varepsilon > 0$  and access to an unknown graph  $G = ([n], E)$ , output “accept” with probability at least  $\frac{2}{3}$  if  $G$  has property  $\mathcal{P}$ , and output “reject” with probability at least  $\frac{2}{3}$  if  $G$  is  $\varepsilon$ -far from having property  $\mathcal{P}$ . The access to  $G$  is given by the following oracle queries: given a query set  $L \subseteq [n]$ , the oracle samples an edge  $(i, j) \sim E$  uniformly at random and returns  $\{i, j\} \cap L$ .

We measure the complexity of algorithms with rejection sampling queries by considering the sizes of the queries. The complexity of an algorithm making queries  $L_1, \dots, L_t \subseteq [n]$  is  $\sum_{i=1}^t |L_i|$ .

The rejection sampling model allows us to study testers which rely on random sampling of edges, while providing the flexibility of making lower-cost queries. This type of query access strikes a delicate balance between simplicity and generality: queries are constrained enough for us to show high lower bounds, and at the same time, the flexibility of making queries allows us to reduce the rejection sampling model to Boolean function testing problems. Specifically, we reduce to tolerant junta testing and tolerant unateness testing (see Subsection 1.1).

Our main result in the rejection sampling model is regarding *non-adaptive* algorithms. These algorithms need to fix their queries in advance and are not allowed to depend on answers to previous queries (in the latter case we say that the algorithm is *adaptive*). We show a lower bound on the complexity of testing whether an unknown graph  $G$  is bipartite using non-adaptive queries.

► **Theorem 1.** *There exists a constant  $\varepsilon > 0$  such that any non-adaptive  $\varepsilon$ -tester for bipartiteness in the rejection sampling model has cost  $\tilde{\Omega}(n^2)$ .*<sup>4</sup>

More specifically, Theorem 1 follows from applying Yao’s principle to the following lemma.

► **Lemma 2.** *Let  $\mathcal{G}_1$  be the uniform distribution over the union of two disjoint cliques of size  $n/2$ , and let  $\mathcal{G}_2$  be the uniform distribution over complete bipartite graphs with each part of size  $n/2$ . Any deterministic non-adaptive algorithm that can distinguish between  $\mathcal{G}_1$  and  $\mathcal{G}_2$  with constant probability using rejection sampling queries, must have complexity  $\tilde{\Omega}(n^2)$ .*

We discuss a number of applications of the rejection sampling model (specifically, of Lemma 2) in the next subsection. In particular, we obtain new lower bounds in the *tolerant testing framework* introduced by Parnas, Ron, and Rubinfeld in [34] for two well-studied properties of Boolean functions (specifically,  $k$ -juntas and unateness; see the next subsection for definitions of these properties). These lower bounds are obtained by a reduction from the rejection sampling model; we show that too-good-to-be-true Boolean function testers for these properties imply the existence of rejection sampling algorithms which distinguish  $\mathcal{G}_1$  and  $\mathcal{G}_2$  with  $\tilde{o}(n^2)$  complexity. Therefore, we may view the rejection sampling model as a useful abstraction in studying the hard instances of tolerant testing  $k$ -juntas and unateness.

<sup>3</sup> The distance definition can be modified accordingly when one considers bounded degree or sparse graphs.

<sup>4</sup> We use the notations  $\tilde{O}, \tilde{\Omega}$  to hide polylogarithmic dependencies on the argument, i.e. for expressions of the form  $O(f \log^c f)$  and  $\Omega(f / \log^c f)$  respectively (for some absolute constant  $c$ ).

## 1.1 Applications to Tolerant Testing: Juntas and Unateness

Given  $n \in \mathbb{N}$  and a subset  $\mathcal{P}$  of  $n$ -variable Boolean functions, a Boolean function  $f: \{0, 1\}^n \rightarrow \{0, 1\}$  has property  $\mathcal{P}$  if  $f \in \mathcal{P}$ . The distance between Boolean functions  $f, g: \{0, 1\}^n \rightarrow \{0, 1\}$  is  $\text{dist}(f, g) = \Pr_{x \sim \{0, 1\}^n}[f(x) \neq g(x)]$ . The distance of  $f$  to the property  $\mathcal{P}$  is  $\text{dist}(f, \mathcal{P}) = \min_{g \in \mathcal{P}} \text{dist}(f, g)$ . We say that  $f$  is  $\varepsilon$ -close to  $\mathcal{P}$  if  $\text{dist}(f, \mathcal{P}) \leq \varepsilon$  and  $f$  is  $\varepsilon$ -far from  $\mathcal{P}$  if  $\text{dist}(f, \mathcal{P}) > \varepsilon$ . The problem of *tolerant property testing* [34] of  $\mathcal{P}$  asks for query-efficient randomized algorithms for the following task:

Given parameters  $0 \leq \varepsilon_0 < \varepsilon_1 < 1$  and black-box query access to a Boolean function  $f: \{0, 1\}^n \rightarrow \{0, 1\}$ , accept with probability at least  $\frac{2}{3}$  if  $f$  is  $\varepsilon_0$ -close to  $\mathcal{P}$  and reject with probability at least  $\frac{2}{3}$  if  $f$  is  $\varepsilon_1$ -far from  $\mathcal{P}$ .

An algorithm which performs the above task is an  $(\varepsilon_0, \varepsilon_1)$ -tolerant tester for  $\mathcal{P}$ . A  $(0, \varepsilon_1)$ -tolerant tester is a *standard* property tester or a *non-tolerant* tester. As noted in [34], tolerant testing is not only a natural generalization, but is also very often the desirable attribute of testing algorithms. This motivates the high level question: how does the requirement of being tolerant affect the complexity of testing the properties studied? We make progress on this question by showing query-complexity separations for two well-studied properties of Boolean functions:  $k$ -juntas, and unate functions.

- ( $k$ -junta) A function  $f: \{0, 1\}^n \rightarrow \{0, 1\}$  is a  $k$ -junta if it depends on at most  $k$  of its variables, i.e., there exists  $k$  distinct indices  $i_1, \dots, i_k \in [n]$  and a  $k$ -variable function  $g: \{0, 1\}^k \rightarrow \{0, 1\}$  where  $f(x) = g(x_{i_1}, \dots, x_{i_k})$  for all  $x \in \{0, 1\}^n$ .
- (unateness) A function  $f: \{0, 1\}^n \rightarrow \{0, 1\}$  is *unate* if  $f$  is either non-increasing or non-decreasing in every variable. Namely, there exists a string  $r \in \{0, 1\}^n$  such that the function  $f(x \oplus r)$  is monotone with respect to the bit-wise partial order on  $\{0, 1\}^n$ .

While separations between tolerant and non-tolerant testing of Boolean function properties were known for an (artificial) property (see Subsection 1.2), these results are the first to give such lower bounds for a natural class of well-studied properties of Boolean functions. The first such theorem we state concerns non-adaptive tolerant testers for  $k$ -juntas.

► **Theorem 3.** *For any  $\alpha < 1$ , there exists constants  $0 < \varepsilon_0 < \varepsilon_1 < 1$  such that for any  $k = k(n) \leq \alpha n$ , any non-adaptive  $(\varepsilon_0, \varepsilon_1)$ -tolerant  $k$ -junta tester must make  $\tilde{\Omega}(k^2)$  queries.*

We give a noteworthy consequences of the Theorem 3. In [7], Blais gave a non-adaptive  $\tilde{O}(k^{3/2})$ -query tester for (non-tolerant) testing of  $k$ -juntas, which was shown to be optimal for non-adaptive algorithms by Chen, Servedio, Tan, Waingarten and Xie in [17]. Combined with Theorem 3, this shows a polynomial separation in the query complexity of non-adaptive tolerant junta testing and non-adaptive junta testing.

The next two theorems concern tolerant testers for unateness.

► **Theorem 4.** *There exists constants  $0 < \varepsilon_0 < \varepsilon_1 < 1$  such that any (possibly adaptive)  $(\varepsilon_0, \varepsilon_1)$ -tolerant unateness tester must make  $\tilde{\Omega}(n)$  queries.*

► **Theorem 5.** *There exists constant  $0 < \varepsilon_0 < \varepsilon_1 < 1$  such that any non-adaptive  $(\varepsilon_0, \varepsilon_1)$ -tolerant unateness tester must make  $\tilde{\Omega}(n^{3/2})$  queries.*

A similar separation in tolerant and non-tolerant testing occurs for the property of unateness as a consequence of Theorem 4 and Theorem 5. Recently, in [3], Baleshzar, Chakrabarty, Pallavoor, Raskhodnikova, and Seshadhri gave a non-adaptive  $\tilde{O}(n)$ -query tester for (non-tolerant) unateness testing, and Chen, Waingarten and Xie [18] gave an

(adaptive)  $\tilde{O}(n^{3/4})$ -query tester for (non-tolerant) unateness testing. We thus, conclude that by Theorem 4 and Theorem 5, tolerant unateness testing is polynomially harder than (non-tolerant) unateness testing, in both adaptive and non-adaptive settings.

## 1.2 Related Work

The properties of  $k$ -juntas and unateness have received much attention in property testing research ([24, 20, 7, 8, 10, 37, 17, 9] study  $k$ -juntas, and [27, 31, 14, 3, 18, 19] study unateness). We briefly review the current state of affairs in (non-tolerant)  $k$ -junta testing and unateness testing, and then discuss tolerant testing of Boolean functions and the rejection sampling model.

**Testing  $k$ -juntas.** The problem of testing  $k$ -juntas, introduced by Fischer, Kindler, Ron, Safra, and Samorodnitsky [24], is now well understood up to poly-logarithmic factors. Chockler and Gutfreund [20] show that any tester for  $k$ -juntas requires  $\Omega(k)$  queries (for a constant  $\varepsilon_1$ ). Blais [8] gave a junta tester that uses  $O(k \log k + k/\varepsilon_1)$  queries, matching the bound of [20] up to a factor of  $O(\log k)$  for constant  $\varepsilon_1$ . When restricted to non-adaptive algorithms, [24] gave a non-adaptive tester making  $\tilde{O}(k^2/\varepsilon_1)$  queries, which was subsequently improved in [7] to  $\tilde{O}(k^{3/2})/\varepsilon_1$ . In terms of lower bounds, Buhrman, Garcia-Soriano, Matsliah, and de Wolf [10] gave a  $\Omega(k \log k)$  lower bound for  $\varepsilon = \Omega(1)$ , and Servedio, Tan, and Wright [37] gave a lower bound which showed a separation between adaptive and non-adaptive algorithms for  $\varepsilon_1 = \frac{1}{\log k}$ . These results were recently improved in [17] to  $\tilde{\Omega}(k^{3/2}/\varepsilon_1)$ , settling the non-adaptive query complexity of the problem up to poly-logarithmic factors.

**Testing unateness.** The problem of testing unateness was introduced alongside the problem of testing monotonicity in Goldreich, Goldwasser, Lehman, Ron, and Samorodnitsky [27], where they gave the first  $O(n^{3/2}/\varepsilon_1)$ -query non-adaptive tester. Khot and Shinkar [31] gave the first improvement by giving a  $\tilde{O}(n/\varepsilon_1)$ -query adaptive algorithm. A non-adaptive algorithm with  $\tilde{O}(n/\varepsilon_1)$  queries was given in [13, 3]. Recently, [18, 2] show that  $\tilde{\Omega}(n)$  queries are necessary for non-adaptive one-sided testers. Subsequently, [19] gave an adaptive algorithm testing unateness with query complexity  $\tilde{O}(n^{3/4}/\varepsilon_1^2)$ . The current best lower bound for general adaptive testers appears in [18], where it was shown that any adaptive two-sided tester must use  $\tilde{\Omega}(n^{2/3})$  queries.

**Tolerant testing.** Once we consider tolerant testing, i.e., the case  $\varepsilon_0 > 0$ , the picture is not as clear. In the paper introducing tolerant testing, [34] observed that standard algorithms whose queries are uniform (but not necessarily independent) are inherently tolerant to some extent. Nevertheless, achieving  $(\varepsilon_0, \varepsilon_1)$ -tolerant testers for constants  $0 < \varepsilon_0 < \varepsilon_1$ , can require applying different methods and techniques (see e.g, [30, 34, 25, 1, 32, 33, 22, 11, 6, 5, 38]).

By applying the observation from [34] to the unateness tester in [3], the tester accepts functions which are  $O(\varepsilon_1/n)$ -close to unate with constant probability. We similarly obtain weak guarantees for tolerant testing of  $k$ -juntas. Diakonikolas, Lee, Matulef, Onak, Rubinfeld, Servedio, and Wan [21] observed that one of the (non-adaptive) junta testers from [24] accepts functions that are  $\text{poly}(\varepsilon_1, 1/k)$ -close to  $k$ -juntas. Chakraborty, Fischer, Garcia-Soriano, and Matsliah [15] noted that the analysis of the junta tester of Blais [8] implicitly implies an  $\exp(k/\varepsilon_1)$ -query complexity tolerant tester which accepts functions that are  $\varepsilon_1/c$ -close to some  $k$ -junta (for some constant  $c > 1$ ) and rejects functions that are  $\varepsilon_1$ -far from every  $k$ -junta. Recently, Blais, Canonne, Eden, Levi and Ron [9] showed that when required to distinguish between the cases that  $f$  is  $\varepsilon_1/10$ -close to a  $k$ -junta, or is  $\varepsilon_1$ -far from a  $2k$ -junta,  $\text{poly}(k, 1/\varepsilon_1)$  queries suffice.



For general properties of Boolean functions, tolerant testing could be much harder than standard testing. Fischer and Fortnow [23] used PCPs in order to construct a property of Boolean functions  $\mathcal{P}$  which is  $(0, \varepsilon_1)$ -testable with a constant number of queries (depending on  $\varepsilon_1$ ), but any  $(1/4, \varepsilon_1)$ -tolerant test for  $\mathcal{P}$  requires  $n^c$  queries for some  $c > 0$ . While [23] presents a strong separation between tolerant and non-tolerant testing, the complexity of tolerant testing of many natural properties remains open. We currently neither have a  $\text{poly}(k, \frac{1}{\varepsilon_1})$ -query tester which  $(\varepsilon_0, \varepsilon_1)$ -tests  $k$ -juntas, nor a  $\text{poly}(n, \frac{1}{\varepsilon_1})$ -query tester that  $(\varepsilon_0, \varepsilon_1)$ -tests unateness or monotonicity when  $\varepsilon_0 = \Theta(\varepsilon_1)$ .

**Testing graphs with rejection sampling queries.** Even though the problem of testing graphs with rejection sampling queries has not been previously studied, the model shares characteristics with previous studied frameworks. These include sample-based testing studied by Goldreich, Goldwasser, and Ron in [28, 29], where the oracle receives random samples from the input. One crucial difference between rejection sampling algorithms (which always query  $[n]$ ) and sample-based testers is the fact that rejection sampling algorithms only receive *positive* examples (in the form of edges), as opposed to random positions in the adjacency matrix (which may be a *negative* example indicated the non-existence of an edge).

The rejection sampling model for graph testing also bears some resemblance to the conditional sampling framework for distribution testing introduced in Canonne, Ron, and Servedio, as well as Chakraborty, Fischer, Goldhirsh, and Matsliah [12, 16], where the algorithm specifies a query set and receives a sample conditioned on it lying in the query set.

### 1.3 Techniques and High Level Overview

We first give an overview of how the lower bound in the rejection sampling model (Lemma 2) implies lower bounds for tolerant testing of  $k$ -juntas and unateness, and then we give an overview of how Lemma 2 is proved.

**Reducing Boolean Function Testing from Rejection Sampling.** This work should be considered alongside some recent works showing lower bounds for testing the properties of monotonicity, unateness, and juntas in the standard property testing model [4, 18, 17]. At a high level, the lower bounds for Boolean function testing proceed in three steps:

1. First, design a randomized indexing function  $\Gamma: \{0, 1\}^n \rightarrow [N]$  that partitions the Boolean cube  $\{0, 1\}^n$  into roughly equal parts in a way compatible with the property (either junta, or unateness). We want to ensure that algorithms that make few queries cannot learn too much about  $\Gamma$ , and that queries falling in the same part are close in Hamming distance.
2. Second, define two distributions over functions  $\mathbf{h}_i: \{0, 1\}^n \rightarrow \{0, 1\}$  for each  $i \in [N]$ . The hard functions are defined by  $\mathbf{f}(x) = \mathbf{h}_{\Gamma(x)}(x)$ , so that one distribution corresponds to functions with the property, and the other distribution corresponds to functions far from the property.
3. Third, show that any testing algorithm for the property is actually solving some algorithmic task (determined by the distributions of  $\mathbf{h}_i$ ) which is hard when queries are close in Hamming distance.

The first step in the above-mentioned plan is standard (given familiarity with [18] and [17]). We will use a construction from [17] for the junta lower bound and a Talagrand-based construction (similar to [18], but somewhat simpler) for the unateness lower bounds. The novelty in this work lies in steps 2 and 3. We will define the distributions over sub-functions  $\mathbf{h}_i$  such that the resulting Boolean functions  $\mathbf{f}(x) = \mathbf{h}_{\Gamma(x)}(x)$  either is  $\varepsilon_0$ -close to desired

property ( $k$ -juntas and unateness), or is  $\varepsilon_1$ -far from having the desired property ( $k$ -juntas and unateness). Then, we will show that any algorithm for tolerant testing of  $k$ -juntas or unateness must be able to solve a hard instance of bipartiteness testing in the rejection sampling model.

At a very high level, our reductions will follow by associating to each distribution of Boolean functions  $\mathbf{f}: \{0, 1\}^n \rightarrow \{0, 1\}$  a distribution over graphs  $\mathbf{G}$  defined on a subset of  $[n]$  (these will be  $\mathcal{G}_1$  and  $\mathcal{G}_2$ ). The edges of a graph  $\mathbf{G}$  sampled from  $\mathcal{G}_1$  or  $\mathcal{G}_2$  will encode how the variables of  $\mathbf{f}$  interact with one another, and the distance of  $\mathbf{f}$  to  $k$ -junta (or unateness) will depend on a global parameter of the  $\mathbf{G}$ .<sup>5</sup> In addition, Boolean function queries on  $\mathbf{f}$  will be interpreted as rejection sampling queries to  $\mathbf{G}$ , so that tests distinguishing the distributions of Boolean functions will give rise to rejection sampling algorithms which distinguish between  $\mathcal{G}_1$  and  $\mathcal{G}_2$ . Since we will show a lower bound in the rejection sampling model, we will obtain a lower bound for tolerant testing of  $k$ -juntas and unateness.

For a more detailed discussion of the distributions and the reductions see Sections 3 and 4.

**Distinguishing  $\mathcal{G}_1$  and  $\mathcal{G}_2$  with Rejection Sampling Queries.** In order to prove Lemma 2, one needs to rule out any deterministic non-adaptive algorithm which distinguishes between  $\mathcal{G}_1$  and  $\mathcal{G}_2$  with rejection sampling queries of complexity  $\tilde{o}(n^2)$ . In order to keep the discussion at a high level, we identify three possible “strategies” for determining whether an underlying graph is a complete bipartite graph, or a union of two disjoint cliques:

1. One approach is for the algorithm to sample edges and consider the subgraph obtained from edges returned by the oracle. For instance, the algorithm may make all rejection sampling queries to be  $[n]$ . These queries are expensive in the rejection sampling model, but they guarantee that an edge from the graph will be observed. If the algorithm is lucky, and there exists a triangle in the subgraph observed, the graph must not be bipartite, so it must come from  $\mathcal{G}_2$ .
2. Another sensible approach is for the algorithm to forget about the structure of the graph, and simply view the distribution on the edges generated by the randomness in the rejection sampling oracle as a distribution testing problem. Suppose for simplicity that the algorithm makes rejection sampling queries  $[n]$ . Then, the corresponding distributions supported on edges from  $\mathcal{G}_1$  and  $\mathcal{G}_2$  will be  $\Omega(1)$ -far from each other, so a distribution testing algorithm can be used.
3. A third, more subtle, approach is for the algorithm to use the fact that  $\mathcal{G}_1$  and  $\mathcal{G}_2$  correspond to the union of two cliques and a complete bipartite graph, and extract knowledge about the non-existence of edges when making queries which return either  $\emptyset$  or a single vertex. More specifically, suppose that by having observed some edges, the algorithm observes two connected components  $L_1$  and  $L_2$ . If when querying  $L_1 \cup L_2$  multiple times, we do not observe an edge, it is more likely the underlying graph comes from  $\mathcal{G}_1$  than  $\mathcal{G}_2$ . Specifically, if  $\mathbf{G} \sim \mathcal{G}_1$  and  $L_1$  lies in one clique and  $L_2$  lies in the other clique, there would be no edges with edges from  $L_1$  and  $L_2$ ; on the other hand, if  $\mathbf{G} \sim \mathcal{G}_2$ , then  $L_1$  and  $L_2$  would always have some edges between them.

---

<sup>5</sup> The relevant graph parameter in  $k$ -juntas and unateness will be different. Luckily, both graph parameters will have gaps in their value depending on the distribution the graphs were drawn from (either  $\mathcal{G}_1$  or  $\mathcal{G}_2$ ). This allows us to reuse the work of proving Lemma 2 to obtain Theorem 3, Theorem 4, and Theorem 5.

The three strategies mentioned above all fail to give  $\tilde{O}(n^2)$  rejection sampling algorithms. The first approach fails because with a budget of  $\tilde{O}(n^2)$ , rejection sampling algorithms will observe subgraphs which consist of various trees of size at most  $\log n$ , thus we will not observe cycles. The second approach fails since the distributions are supported on  $\Omega(n^2)$  edges, so distribution testing algorithms will require  $\Omega(n)$  edges (which costs  $\Omega(n^2)$ ) to distinguish between  $\mathcal{G}_1$  and  $\mathcal{G}_2$ . Finally, the third approach fails since algorithms will only observe  $o(n)$  responses from the oracle corresponding to lone vertices which will be split roughly evenly among the unknown parts of the graph, so these observations will not be enough to distinguish between  $\mathcal{G}_1$  and  $\mathcal{G}_2$ .

Our lower bound rules out the three strategies sketched above when the complexity is  $\tilde{O}(n^2)$ , and shows that if the above three strategies do not work (in any possible combination with each other as well), then no non-adaptive algorithm of complexity  $\tilde{O}(n^2)$  will work. The main technical challenge is to show that the above strategies are the *only* possible strategies to distinguish  $\mathcal{G}_1$  and  $\mathcal{G}_2$ . In Section 5, we give a more detailed, yet still high-level discussion of the proof of Lemma 2.

Finally, the analysis of Lemma 2 is tight; there is a non-adaptive rejection sampling algorithm which distinguishes  $\mathcal{G}_1$  and  $\mathcal{G}_2$  with complexity  $\tilde{O}(n^2)$ . The algorithm (based on the first approach mentioned above) is simple: make  $\tilde{O}(n)$  queries  $L = [n]$ , and if we observe an odd-length cycle, we output “ $\mathcal{G}_1$ ”, otherwise, output “ $\mathcal{G}_2$ ”.

## 1.4 Preliminaries

We use boldfaced letters such as  $\mathbf{A}, \mathbf{M}$  to denote random variables. Given a string  $x \in \{0, 1\}^n$  and  $j \in [n]$ , we write  $x^{(j)}$  to denote the string obtained from  $x$  by flipping the  $j$ -th coordinate. An edge along the  $j$ -th direction in  $\{0, 1\}^n$  is a pair  $(x, y)$  of strings with  $y = x^{(j)}$ . In addition, for  $\alpha \in \{0, 1\}$  we use the notation  $x^{(j \rightarrow \alpha)}$  to denote the string  $x$  where the  $j$ th coordinate is set to  $\alpha$ . Given  $x \in \{0, 1\}^n$  and  $S \subseteq [n]$ , we use  $x|_S \in \{0, 1\}^S$  to denote the projection of  $x$  on  $S$ . For a distribution  $\mathcal{D}$  we write  $\mathbf{d} \sim \mathcal{D}$  to denote an element  $d$  drawn according to the distribution. We sometimes write  $a \approx b \pm c$  to denote  $b - c \leq a \leq b + c$ .

## 2 The Rejection Sampling Model

In this section, we define the rejection sampling model and the distributions over graphs we will use throughout this work. We define the rejection sampling model tailored to our specific application of proving Lemma 2.

► **Definition 6.** Consider two distributions,  $\mathcal{G}_1$  and  $\mathcal{G}_2$  supported on graphs with vertex set  $[n]$ . The problem of distinguishing  $\mathcal{G}_1$  and  $\mathcal{G}_2$  with a rejection sampling oracle aims to distinguish between the following two cases with a specific kind of query:

- Cases: We have an unknown graph  $\mathbf{G} \sim \mathcal{G}_1$  or  $\mathbf{G} \sim \mathcal{G}_2$ .
- Rejection Sampling Oracle: Each query is a subset  $L \subseteq [n]$ ; an oracle samples an edge  $(j_1, j_2)$  from  $\mathbf{G}$  uniformly at random, and the oracle returns  $\mathbf{v} = \{j_1, j_2\} \cap L$ . The complexity of a query  $L$  is given by  $|L|$ .

We say a non-adaptive algorithm Alg for this problem is a sequence of query sets  $L_1, \dots, L_q \subseteq [n]$ , as well as a function Alg:  $([n] \cup ([n] \times [n]) \cup \{\emptyset\})^q \rightarrow \{\text{“}\mathcal{G}_1\text{”}, \text{“}\mathcal{G}_2\text{”}\}$ . The algorithm sends each query to the oracle, and for each query  $L_i$ , the oracle responds  $\mathbf{v}_i \in [n] \cup ([n] \times [n]) \cup \{\emptyset\}$ , which is either a single element of  $[n]$ , an edge in  $\mathbf{G}$ , or  $\emptyset$ . The algorithm succeeds if:

$$\Pr_{\substack{\mathbf{G} \sim \mathcal{G}_1, \\ \mathbf{v}_1, \dots, \mathbf{v}_q}} [\text{Alg}(\mathbf{v}_1, \dots, \mathbf{v}_q) \text{ outputs “}\mathcal{G}_1\text{”}] - \Pr_{\substack{\mathbf{G} \sim \mathcal{G}_2, \\ \mathbf{v}_1, \dots, \mathbf{v}_q}} [\text{Alg}(\mathbf{v}_1, \dots, \mathbf{v}_q) \text{ outputs “}\mathcal{G}_1\text{”}] \geq \frac{1}{3}.$$

The complexity of Alg is measured by the sum of the complexity of the queries, so we let  $\text{cost}(\text{Alg}) = \sum_{i=1}^q |L_i|$ .

While our interest in this work is primarily on lower bounds for the rejection sampling model, an interesting direction is to explore upper bounds of various natural graph properties with rejection sampling queries. Our specific applications only require ruling out non-adaptive algorithms, but one may define adaptive algorithms in the rejection sampling model and study the power of adaptivity in this setting as well.

## 2.1 The Distributions $\mathcal{G}_1$ and $\mathcal{G}_2$

Let  $\mathcal{G}_1$  and  $\mathcal{G}_2$  be two distributions supported on graphs with vertex set  $[n]$  defined as follows. Let  $\mathbf{A} \subseteq [n]$  be a uniform random subset of size  $\frac{n}{2}$ .

$$\mathcal{G}_1 = \left\{ K_{\mathbf{A}} \cup K_{\overline{\mathbf{A}}} : \mathbf{A} \subseteq [n] \text{ random subset size } \frac{n}{2} \right\}$$

$$\mathcal{G}_2 = \left\{ K_{\mathbf{A}, \overline{\mathbf{A}}} : \mathbf{A} \subseteq [n] \text{ random subset size } \frac{n}{2} \right\},$$

where for a subset  $A$ ,  $K_A$  is the complete graph on vertices in  $A$  and  $K_{A, \overline{A}}$  is the complete bipartite graph whose sides are  $A$  and  $\overline{A}$ .

## 3 Tolerant Junta Testing

In this section, we will prove that distinguishing the two distributions  $\mathcal{G}_1$  and  $\mathcal{G}_2$  using a rejection sampling oracle reduces to distinguishing two distributions  $\mathcal{D}_{\text{yes}}$  and  $\mathcal{D}_{\text{no}}$  over Boolean functions, where  $\mathcal{D}_{\text{yes}}$  is supported on functions that are close to  $k$ -juntas and  $\mathcal{D}_{\text{no}}$  is supported on functions that are far from any  $k$ -junta with high probability.

### 3.1 High Level Overview

We start by providing some intuition of how our constructions and reduction implement the plan set forth in Subsection 1.3 for the property of being a  $k$ -junta. We define two distributions supported on Boolean functions,  $\mathcal{D}_{\text{yes}}$  and  $\mathcal{D}_{\text{no}}$ , so that functions in  $\mathcal{D}_{\text{yes}}$  are  $\varepsilon_0$ -close to being  $k$ -juntas and functions in  $\mathcal{D}_{\text{no}}$  are  $\varepsilon_1$ -far from being  $k$ -juntas (where  $\varepsilon_0$  and  $\varepsilon_1$  are appropriately defined constants and  $k = \frac{3n}{4}$ ).

As mentioned in the introduction, our distributions are based on the indexing function used in [17]. We draw a uniform random subset  $\mathbf{M} \subseteq [n]$  of size  $n/2$  and our function  $\Gamma = \Gamma_{\mathbf{M}}: \{0, 1\}^n \rightarrow \{0, 1\}^{[2^{n/2}]}$  projects the points onto the variables in  $\mathbf{M}$ . Thus, it remains to define the sequence of functions  $\mathbf{H} = (\mathbf{h}_i: \{0, 1\}^n \rightarrow \{0, 1\} : i \in [2^{n/2}])$ .

We will sample a graph  $\mathbf{G} \sim \mathcal{G}_1$  (in the case of  $\mathcal{D}_{\text{yes}}$ ), and a graph  $\mathbf{G} \sim \mathcal{G}_2$  (in the case of  $\mathcal{D}_{\text{no}}$ ) supported on vertices in  $\overline{\mathbf{M}}$ . Each function  $\mathbf{h}_i: \{0, 1\}^n \rightarrow \{0, 1\}$  is given by first sampling an edge  $(j_1, j_2) \sim \mathbf{G}$  and letting  $\mathbf{h}_i$  be a parity (or a negated parity) of the variables  $x_{j_1}$  and  $x_{j_2}$ . Thus, a function  $\mathbf{f}$  from  $\mathcal{D}_{\text{yes}}$  or  $\mathcal{D}_{\text{no}}$  will have all variables being relevant, however, we will see that functions in  $\mathcal{D}_{\text{yes}}$  have a group of  $\frac{n}{4}$  variables which can be eliminated efficiently<sup>6</sup>.

We think of the sub-functions  $\mathbf{h}_i$  defined with respect to edges from  $\mathbf{G}$  as implementing a sort of *gadget*: the gadget defined with respect to an edge  $(j_1, j_2)$  will have the property that if  $\mathbf{f}$  eliminates the variable  $j_1$ , it will be “encouraged” to eliminate variable  $j_2$  as well.

<sup>6</sup> We say that a variable is eliminated if we change the function to remove the dependence of the variable.

In fact, each time an edge  $(j_1, j_2) \sim \mathbf{G}$  is used to define a sub-function  $h_i$ , any  $k$ -junta  $g: \{0, 1\}^n \rightarrow \{0, 1\}$  where variable  $j_1$  or  $j_2$  is irrelevant will have to change half of the corresponding part indexed by  $\Gamma$ . Intuitively, a function  $f \sim \mathcal{D}_{\text{yes}}$  or  $\mathcal{D}_{\text{no}}$  (which originally depends on all  $n$  variables) wants to eliminate its dependence of  $n - k$  variables in order to become a  $k$ -junta. When  $f$  picks a variable  $j \in \overline{\mathbf{M}}$  to eliminate (since variables in  $\mathbf{M}$  are too expensive), it must change points in parts where the edge sampled is incident on  $j$ . The key observation is that when  $f$  needs to eliminate multiple variables, if  $f$  picks the variables  $j_1$  and  $j_2$  to eliminate, whenever a part samples the edge  $(j_1, j_2)$ , the function changes the points in one part and eliminates two variables. Thus,  $f$  eliminates two variables by changing the same number of points when there are edges between  $j_1$  and  $j_2$ .

At a high level, the gadgets encourage the function  $f$  to remove the dependence of variables within a group of edges, i.e., the closest  $k$ -junta will correspond to a function  $g$  which eliminates groups of variables with edges within each other and few outgoing edges. More specifically, if we want to eliminate  $\frac{n}{4}$  variables from  $f$ , we must find a bisection of the graph  $\mathbf{G}$  whose cut value is small; in the case of  $\mathcal{G}_1$ , one of the cliques will have cut value 0, whereas any bisection of a graph from  $\mathcal{G}_2$  will have a high cut value, which makes functions in  $\mathcal{D}_{\text{yes}}$  closer to  $\frac{3n}{4}$ -juntas than functions in  $\mathcal{D}_{\text{no}}$ .

The reduction from rejection sampling is straight-forward. We consider all queries which are indexed to the same part, and if two queries indexed to the same part differ on a variable  $j$ , then the algorithm “explores” direction  $j$ . Each part  $i \in [2^{n/2}]$  where some query falls in has a corresponding rejection sampling query  $L_i$ , which queries the variables explored by the Boolean function testing algorithm.

### 3.2 The Distributions $\mathcal{D}_{\text{yes}}$ and $\mathcal{D}_{\text{no}}$

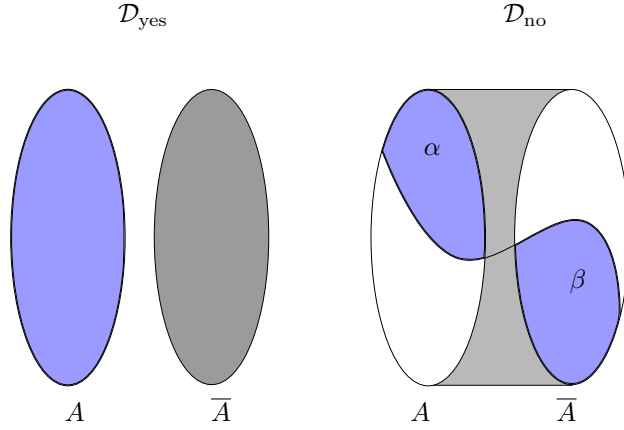
The goal of this subsection is to define the two distributions  $\mathcal{D}_{\text{yes}}$  and  $\mathcal{D}_{\text{no}}$ , supported over Boolean functions with  $n$  variables. Functions  $f \in \mathcal{D}_{\text{yes}}$  will be *close* to being a  $k$ -junta with high probability, and functions  $f \sim \mathcal{D}_{\text{no}}$  will be *far* from any  $k$ -junta with high probability. We note that it suffices to consider  $k = \frac{3n}{4}$  to obtain Theorem 3. We refer the reader to the full version of the paper for the reduction from arbitrary  $k$  to  $k = \frac{3n}{4}$ .

**Distribution  $\mathcal{D}_{\text{yes}}$ .** A function  $f$  from  $\mathcal{D}_{\text{yes}}$  is generated from a tuple of three random variables,  $(\mathbf{M}, \mathbf{A}, \mathbf{H})$ , and we set  $f = f_{\mathbf{M}, \mathbf{A}, \mathbf{H}}$ . The tuple is drawn according to the following randomized procedure:

1. Sample a uniformly random subset  $\mathbf{M} \subseteq [n]$  of size  $m \stackrel{\text{def}}{=} \frac{n}{2}$ . Let  $N = 2^m$  and  $\Gamma_{\mathbf{M}}: \{0, 1\}^n \rightarrow [N]$  be the function that maps  $x \in \{0, 1\}^n$  to a number encoded by  $x|_{\mathbf{M}} \in [N]$ .
2. Sample  $\mathbf{A} \subseteq \overline{\mathbf{M}}$  of size  $\frac{n}{4}$  uniformly at random, and consider the graph  $\mathbf{G}$  defined on vertices  $[\overline{\mathbf{M}}]$  with  $\mathbf{G} = K_{\mathbf{A}} \cup K_{\overline{\mathbf{A}}}$ , i.e.,  $\mathbf{G}$  is a uniformly random graph drawn according to  $\mathcal{G}_1$ .
3. Define a sequence of  $N$  functions  $\mathbf{H} = \{h_i: \{0, 1\}^n \rightarrow \{0, 1\} : i \in [N]\}$  drawn from a distribution  $\mathcal{E}(\mathbf{G})$ . For each  $i \in \{1, \dots, N/2\}$ , we let  $h_i(x) = \bigoplus_{\ell \in \mathbf{M}} x_{\ell}$ . For each  $i \in \{N/2 + 1, \dots, N\}$ , we will generate  $h_i$  independently by sampling an edge  $(j_1, j_2) \sim \mathbf{G}$  uniformly at random, as well as a uniform random bit  $r \sim \{0, 1\}$ . We let

$$h_i(x) = x_{j_1} \oplus x_{j_2} \oplus r.$$

4. Using  $\mathbf{M}, \mathbf{A}$  and  $\mathbf{H}$ , define  $f_{\mathbf{M}, \mathbf{A}, \mathbf{H}} = h_{\Gamma_{\mathbf{M}}(x)}(x)$  for each  $x \in \{0, 1\}^n$ .



**Figure 1** Example of graphs  $\mathbf{G}$  from  $\mathcal{D}_{\text{yes}}$  and  $\mathcal{D}_{\text{no}}$ . On the left, the graph  $\mathbf{G}$  is the union of two cliques of size  $\frac{n}{4}$ , corresponding to  $\mathcal{D}_{\text{yes}}$ . We note that  $\chi(G) = \frac{1}{2}$ , since if we let  $S = \mathbf{A}$  (pictured as the blue set), we see that  $S$  contains half of the edges. On the right, the graph  $\mathbf{G}$  is the complete bipartite graph with side sizes  $\frac{n}{4}$ , corresponding to  $\mathcal{D}_{\text{no}}$ . We note that  $\chi(G) = \frac{3}{4}$ : consider any set  $S \subseteq \overline{M}$  of size at least  $\frac{n}{4}$  pictured in the blue region, and let  $\alpha = |S \cap A|$  and  $\beta = |S \cap \overline{A}|$ , where  $\alpha + \beta \geq \frac{n}{4}$ , so  $E(S, S) + E(S, \overline{S}) \geq \binom{n}{4}^2 - \alpha\beta \geq \binom{n}{4}^2(1 - \frac{1}{4})$ .

**Distribution  $\mathcal{D}_{\text{no}}$ .** A function  $f$  drawn from  $\mathcal{D}_{\text{no}}$  is also generated by first drawing the tuple  $(\mathbf{M}, \mathbf{A}, \mathbf{H})$  and setting  $f = f_{\mathbf{M}, \mathbf{A}, \mathbf{H}}$ . Both  $\mathbf{M}$  and  $\mathbf{A}$  are drawn using the same procedure; the only difference is that the graph  $\mathbf{G} = K_{\mathbf{A}, \overline{\mathbf{A}}}$ , i.e.,  $\mathbf{G}$  is a uniformly random graph drawn according to  $\mathcal{G}_2$ . Then  $\mathbf{H} \sim \mathcal{E}(\mathbf{G})$  is sampled from the modified graph  $\mathbf{G}$ .

We let  $k \stackrel{\text{def}}{=} \frac{3n}{4}$ ,  $\varepsilon_0 \stackrel{\text{def}}{=} \frac{1}{8}$ , and  $\varepsilon_1 \stackrel{\text{def}}{=} \frac{3}{16}$ . Consider a fixed subset  $M \subseteq [n]$  which satisfies  $|M| = \frac{n}{2}$ , and a fixed subset  $A \subseteq \overline{M}$  which satisfies  $|A| = \frac{n}{4}$ . Let  $G$  be a graph defined over vertices in  $\overline{M}$ , and for any subsets  $S_1, S_2 \subseteq \overline{M}$ , let  $E_G(S_1, S_2) = |\{(j_1, j_2) \in G : j_1 \in S_1, j_2 \in S_2\}|$ , be the number of edges between sets  $S_1$  and  $S_2$ . Additionally, we let

$$\chi(G) = \min \left\{ \frac{E_G(S, S) + E_G(S, \overline{S})}{E_G(\overline{M}, \overline{M})} : S \subseteq \overline{M}, |S| \geq \frac{n}{4} \right\} \quad (1)$$

be the minimum fraction of edges adjacent to a set  $S$  of size at least  $\frac{n}{4}$ . The following lemma relates the distance of a function  $\mathbf{f} = f_{M, \mathbf{A}, \mathbf{H}}$  where  $\mathbf{H} \sim \mathcal{E}(G)$  to being a  $k$ -junta to  $\chi(G)$ . We then apply this lemma to the graph in  $\mathcal{D}_{\text{yes}}$  and  $\mathcal{D}_{\text{no}}$  to show that functions in  $\mathcal{D}_{\text{yes}}$  are  $\varepsilon_0$ -close to being  $k$ -juntas, and functions in  $\mathcal{D}_{\text{no}}$  are  $\varepsilon_1$ -far from being  $k$ -juntas.

**► Lemma 7.** *Let  $G$  be any graph defined over vertices in  $A$ . If  $\mathbf{f} = f_{M, \mathbf{A}, \mathbf{H}}$ , where  $\mathbf{H} \sim \mathcal{E}(G)$ , then with probability at least  $1 - o(1)$ ,*

$$\frac{1}{4} \cdot \chi(G) - o(1) \leq \text{dist}(\mathbf{f}, k\text{-Junta}) \leq \frac{1}{4} \cdot \chi(G) + o(1).$$

**► Corollary 8.** *We have that  $\mathbf{f} \sim \mathcal{D}_{\text{yes}}$  has  $\text{dist}(\mathbf{f}, k\text{-Junta}) \leq \varepsilon_0 + o(1)$  with probability  $1 - o(1)$ , and that  $\mathbf{f} \sim \mathcal{D}_{\text{no}}$  has  $\text{dist}(\mathbf{f}, k\text{-Junta}) \geq \varepsilon_1 - o(1)$  with probability  $1 - o(1)$ .*

The proof shows that distinguishing the two distributions  $\mathcal{G}_1$  and  $\mathcal{G}_2$  using rejection sampling oracle reduces to distinguishing the two distributions  $\mathcal{D}_{\text{yes}}$  and  $\mathcal{D}_{\text{no}}$ .

► **Lemma 9.** *Suppose there exists a deterministic non-adaptive algorithm Alg making  $q$  queries to Boolean functions  $f: \{0, 1\}^{2n} \rightarrow \{0, 1\}$ . Then, there exists a deterministic non-adaptive algorithm Alg' making rejection sampling queries to an  $n$ -vertex graph such that:*

$$\Pr_{f \sim \mathcal{D}_{\text{yes}}} [\text{Alg}(f) \text{ “accepts”}] = \Pr_{\mathbf{G} \sim \mathcal{G}_1} [\text{Alg}'(\mathbf{G}) \text{ outputs “}\mathcal{G}_1\text{”}], \quad \text{and}$$

$$\Pr_{f \sim \mathcal{D}_{\text{no}}} [\text{Alg}(f) \text{ “accepts”}] = \Pr_{\mathbf{G} \sim \mathcal{G}_2} [\text{Alg}'(\mathbf{G}) \text{ outputs “}\mathcal{G}_1\text{”}].$$

and has  $\text{cost}(\text{Alg}') = O(q \log n)$  with probability  $1 - o(1)$  over the randomness in Alg'.

## 4 Tolerant Unateness Testing

In this section, we show how to reduce distinguishing distributions  $\mathcal{G}_1$  and  $\mathcal{G}_2$  to distinguishing between Boolean functions which are close to unate and Boolean functions which are far from unate. We start with a high level overview of the constructions and reduction, and then proceed to give formal definitions and the reductions for adaptive and non-adaptive tolerant testing.

### 4.1 High Level Overview

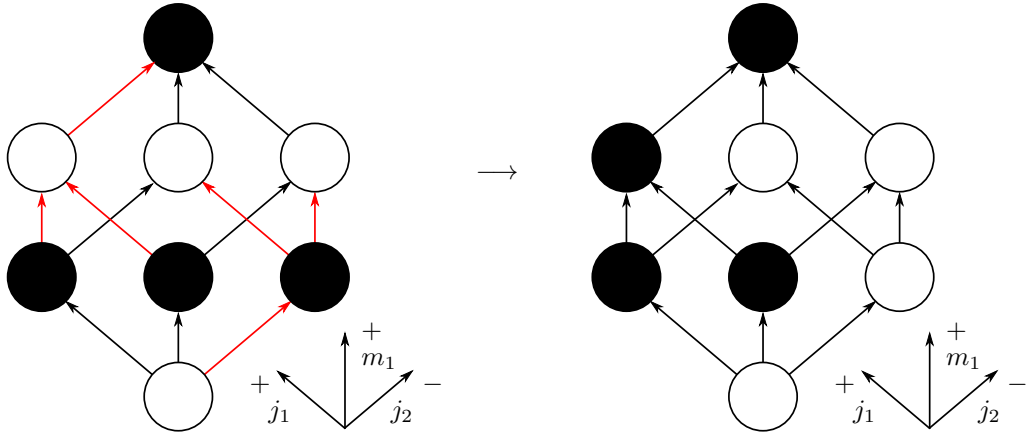
We now describe how our constructions and reduction implement the plan set forth in Subsection 1.3 for the property of unateness. Similarly to Section 3, we define two distributions  $\mathcal{D}_{\text{yes}}$  and  $\mathcal{D}_{\text{no}}$  supported on Boolean functions, so that functions in  $\mathcal{D}_{\text{yes}}$  are  $\varepsilon_0$ -close to being unate, and functions in  $\mathcal{D}_{\text{no}}$  are  $\varepsilon_1$ -far from being unate (where  $\varepsilon_0$  and  $\varepsilon_1$  are appropriately defined constants).

We will use a randomized indexing function  $\Gamma: \{0, 1\}^n \rightarrow [N] \cup \{0^*, 1^*\}$  based on the Talagrand-style constructions from [4, 18] to partition  $\{0, 1\}^n$  in a unate fashion, specifically,  $\Gamma$  will satisfy that for all  $i \neq j \in [N]$ , if  $x, y \in \{0, 1\}^n$  have  $\Gamma(x) = i$  and  $\Gamma(y) = j$ , then  $x$  and  $y$  are *incomparable*,  $x \not\prec y$  and  $y \not\prec x$ . Again, we will then use a graph  $\mathbf{G} \sim \mathcal{G}_1$  or  $\mathcal{G}_2$  to define the sequence of sub-function  $\mathbf{H} = (\mathbf{h}_i: \{0, 1\}^n \rightarrow \{0, 1\} : i \in [N])$ . The sub-functions  $\mathbf{h}_i$  will be given by a parity (or negated parity) of three variables: two variables will correspond to the end points of an edge sampled  $(j_1, j_2) \sim \mathcal{G}$ , the third variable will be one of two pre-specified variables, which we call  $m_1$  and  $m_2$ . Consider for simplicity the case when  $\mathbf{h}_i(x) = x_{j_1} \oplus x_{j_2} \oplus x_{m_1}$ , and assume that we require that variable  $m_1$  is non-decreasing.

Similarly to Section 3, the functions  $\mathbf{h}_i$  are thought of as gadgets. We will have that if  $\mathbf{h}_i$  is defined with respect to an edge  $(j_1, j_2)$  and  $m_1$ , then the function  $\mathbf{f}$  will be “encouraged” to make variables  $j_1$  and  $j_2$  have opposite directions, i.e., either  $j_1$  is non-increasing and  $j_2$  is non-decreasing, or  $j_1$  is non-decreasing and  $j_2$  is non-increasing. In order to see why the three variable parity implements this gadget, we turn our attention to Figure 2 and Figure 3.

Intuitively, the function  $\mathbf{f}$  needs to change some of its inputs to be unate, and it must choose whether the variables  $j_1$  and  $j_2$  will be monotone (non-decreasing) or anti-monotone (non-increasing). Suppose  $\mathbf{f}$  decides that the variable  $j_1$  should be monotone and  $j_2$  be anti-monotone, and  $m_1$  will always be monotone (since it will be too expensive to make it anti-monotone). Then, when  $\mathbf{h}_i(x) = x_{j_1} \oplus x_{j_2} \oplus x_{m_1}$ ,  $\mathbf{h}_i$  will have some *violating edges*, i.e., edges in direction  $j_1$  which are decreasing, or edges in direction  $j_2$  which are increasing, or edges in direction  $m_1$  which are decreasing (see Figure 2, where these violating edges are marked in red). In this case, there exists a way that  $\mathbf{f}$  may change  $\frac{1}{4}$ -th fraction of the points and remove all violating edges (again, this procedure is shown in Figure 2).



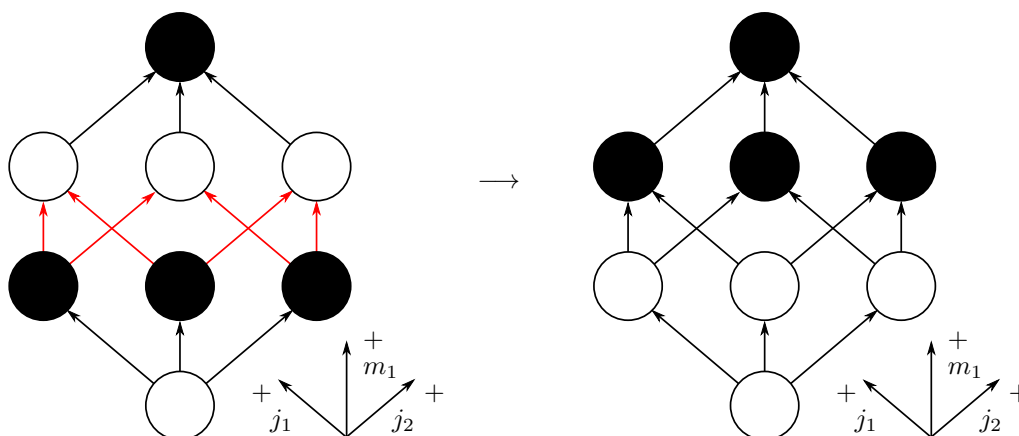


■ **Figure 2** Example of a function  $h_i: \{0, 1\}^n \rightarrow \{0, 1\}$  with  $h_i(x) = x_{j_1} \oplus x_{j_2} \oplus x_{m_1}$  with variable  $j_1$  (which ought to be monotone),  $j_2$  (which ought to be anti-monotone), and  $m_1$  (which is always monotone). The image on the left-hand side represents  $h_i$ , and the red edges correspond to violating edges for variables  $j_1, j_2$  and  $m_1$ . In other words, the red edges correspond to anti-monotone edges in variables  $j_1$ , monotone edges in variables  $j_2$ , and anti-monotone edges in direction  $m_1$ . On the right-hand side, we show how such a function can be “fixed” into a function  $h'_i: \{0, 1\}^n \rightarrow \{0, 1\}$  by changing  $\frac{1}{4}$ -fraction of the points.

In contrast, suppose that  $f$  decides that the variables  $j_1$  and  $j_2$  both should be monotone. Then, when  $h_i(x) = x_{j_1} \oplus x_{j_2} \oplus x_{m_1}$ , the violating edges (shown in Figure 3) form vertex-disjoint cycles of length 6 in  $\{0, 1\}^n$ , thus, the function  $f$  will have to change  $\frac{3}{8}$ -th fraction of the points in order to remove all violating edges. In other words, when there is an edge  $(j_1, j_2)$  sampled in  $h_i$ , the function  $f$  is “encouraged” to make  $j_1$  and  $j_2$  have opposite directions, and “discouraged” to make  $j_1$  and  $j_2$  have the same direction. The other cases are presented in Figures 4, 5, and 6.

In order for  $f$  to become unate, it must first choose whether each variable will be monotone or anti-monotone.  $f$  will choose all variables in  $\mathbf{M}$  to be monotone, the variable  $m_1$  to be monotone, and  $m_2$  to be anti-monotone, but will have to make a choice for each variable in  $\overline{\mathbf{M}}$ , corresponding to each vertex of the graph  $\mathbf{G}$ . As discussed above, for each edge  $(j_1, j_2)$  in the graph,  $f$  is encouraged to make these orientations opposite from each other, so  $f$  will want to look for the maximum cut on the graph, whose value will be different in  $\mathcal{G}_1$  and  $\mathcal{G}_2$ .

Similarly to the case in Section 3, the reduction will follow by defining the rejection sampling queries  $L_i$  corresponding to variables explored in sub-function  $h_i$ . The unate indexing functions  $\Gamma$  are not as strong as the indexing functions from the Section 3, so for each query in the Boolean function testing algorithm, our reduction will lose some cost in the rejection sampling algorithm. In particular, the adaptive reduction loses  $n$  cost for each Boolean function query, since adaptive algorithms can efficiently explore variables with a binary search; this gives the  $\tilde{\Omega}(n)$  lower bound for tolerant unateness testing. The non-adaptive reduction loses  $O(\sqrt{n} \log n)$  cost for each Boolean function query since queries falling in the same part may be  $\Omega(\sqrt{n})$  away from each other (the same scenario occurs in the non-adaptive monotonicity lower bound of [18]). The non-adaptive reduction is more complicated than the adaptive reduction since it is not exactly a black-box reduction (we require a lemma from Section 5). This gives the  $\tilde{\Omega}(n^{3/2})$  lower bound for non-adaptive tolerant unateness testing.



■ **Figure 3** Example of a function  $h_i: \{0, 1\}^n \rightarrow \{0, 1\}$  with  $h_i(x) = x_{j_1} \oplus x_{j_2} \oplus x_{m_1}$  with variables  $j_1$  and  $j_2$  (which ought to be monotone), and  $m_1$  (which ought to be monotone). On the left side, we indicate the violating edges with red arrows, and note that the functions in the left and right differ by  $\frac{3}{8}$ -fraction of the points. We also note that any function  $h'_i: \{0, 1\}^n \rightarrow \{0, 1\}$  which has  $j_1$ ,  $j_2$  and  $m_1$  monotone must differ from  $h_i$  on at least  $\frac{3}{8}$ -fraction of the points because the violating edges of  $h_i$  form a cycle of length 6.

## 4.2 The Distributions $\mathcal{D}_{\text{yes}}$ and $\mathcal{D}_{\text{no}}$

We now turn to describing a pair of distributions  $\mathcal{D}_{\text{yes}}$  and  $\mathcal{D}_{\text{no}}$  supported on Boolean functions  $f: \{0, 1\}^n \rightarrow \{0, 1\}$ . These distributions will have the property that for some constants  $\varepsilon_0$  and  $\varepsilon_1$  with  $0 < \varepsilon_0 < \varepsilon_1$ ,

$$\Pr_{f \sim \mathcal{D}_{\text{yes}}} [\text{dist}(f, \text{Unate}) \leq \varepsilon_0] = 1 - o(1) \quad \text{and} \quad \Pr_{f \sim \mathcal{D}_{\text{no}}} [\text{dist}(f, \text{Unate}) \geq \varepsilon_1] = 1 - o(1).$$

We first define a function  $f \sim \mathcal{D}_{\text{no}}$ , where we fix the parameter  $N = 2^{\sqrt{n}}$ .

1. Sample some set  $\mathbf{M} \subseteq [n]$  of size  $|\mathbf{M}| = \frac{n}{2}$  uniformly at random and let  $m_1, m_2 \sim \mathbf{M}$  be two distinct indices.
2. We let  $\mathbf{T} \sim \mathcal{E}(\mathbf{M} \setminus \{m_1, m_2\})$  (which we describe next).  $\mathbf{T}$  is a sequence of terms  $(\mathbf{T}_i : i \in [N])$  which is used to define a multiplexer map  $\Gamma_{\mathbf{T}}: \{0, 1\}^n \rightarrow [N] \cup \{0^*, 1^*\}$ .
3. We sample  $\mathbf{A} \subseteq \overline{\mathbf{M}}$  of size  $|\mathbf{A}| = \frac{n}{2}$  and define a graph as  $\mathbf{G} = K_{\mathbf{A}} \cup K_{\overline{\mathbf{A}}}$ .
4. We now define the distribution over sub-functions  $\mathbf{H} = (h_i : i \in [N]) \sim \mathcal{H}(m_1, m_2, \mathbf{G})$ . For each function  $h_i: \{0, 1\}^n \rightarrow \{0, 1\}$ , we generate  $h_i$  independently:
  - When  $i \leq 3N/4$ , we sample  $j \sim \{m_1, m_2\}$  and we let:

$$h_i(x) = \begin{cases} x_j & j = m_1 \\ \neg x_j & j = m_2 \end{cases}.$$

- Otherwise, if  $i > 3N/4$ , we sample an edge  $(j_1, j_2) \sim \mathbf{G}$  and an index  $j_3 \sim \{m_1, m_2\}$  we let:

$$h_i(x) = \begin{cases} x_{j_1} \oplus x_{j_2} \oplus x_{j_3} & j_3 = m_1 \\ \neg x_{j_1} \oplus x_{j_2} \oplus x_{j_3} & j_3 = m_2 \end{cases}.$$

52:14 Lower Bounds for Tolerant Junta and Unateness Testing

The function  $\mathbf{f}: \{0, 1\}^n \rightarrow \{0, 1\}$  is given by  $\mathbf{f}(x) = f_{\mathbf{T}, \mathbf{A}, \mathbf{H}}(x)$  where:

$$f_{\mathbf{T}, \mathbf{A}, \mathbf{H}}(x) = \begin{cases} 1 & |x|_{\mathbf{M}} > \frac{n}{4} + \sqrt{n} \\ 0 & |x|_{\mathbf{M}} < \frac{n}{4} - \sqrt{n} \\ 1 & \Gamma_{\mathbf{T}}(x) = 1^* \\ 0 & \Gamma_{\mathbf{T}}(x) = 0^* \\ \mathbf{h}_{\Gamma_{\mathbf{T}}(x)}(x) & \text{otherwise} \end{cases}. \quad (2)$$

We now turn to define the distribution  $\mathcal{E}(M)$  supported on terms  $\mathbf{T}$ , as well as the multiplexer map  $\Gamma_{\mathbf{T}}: \{0, 1\}^n \rightarrow [N]$ . As mentioned above,  $\mathbf{T} \sim \mathcal{E}(M)$  will be a sequence of  $N$  terms  $(\mathbf{T}_i : i \in [N])$ , where each  $\mathbf{T}_i$  is given by a DNF term:  $\mathbf{T}_i(x) = \bigwedge_{j \in \mathbf{T}_i} x_j$ , where the set  $\mathbf{T}_i \subseteq M$  is a uniformly random  $\sqrt{n}$ -element subset. Given the sequence of terms  $\mathbf{T}$ , we let:

$$\Gamma_{\mathbf{T}}(x) = \begin{cases} 0^* & \forall i \in [N], \mathbf{T}_i(x) = 0 \\ 1^* & \exists i_1 \neq i_2 \in [N], \mathbf{T}_{i_1}(x) = \mathbf{T}_{i_2}(x) = 1 \\ i & \mathbf{T}_i(x) = 1 \text{ for a unique } i \in [N] \end{cases}.$$

It remains to define the distribution  $\mathcal{D}_{\text{yes}}$  supported on Boolean functions. The function  $\mathbf{f} \sim \mathcal{D}_{\text{yes}}$  will be defined almost exactly the same. We still have  $\mathbf{f} = f_{\mathbf{T}, \mathbf{A}, \mathbf{H}}$  as defined above, however, the graph  $\mathbf{G}$  will be different. In particular, we will let  $\mathbf{G} = K_{\mathbf{A}, \overline{\mathbf{A}}}$ .

Fix any set  $M \subseteq [n]$  of size  $\frac{n}{2}$  and let  $m_1, m_2 \in M$  be two distinct indices and  $M' = M \setminus \{m_1, m_2\}$ . For any  $\mathbf{T} \sim \mathcal{E}(M')$ , let  $\mathbf{X} \subseteq \{0, 1\}^n$  be the subset of points indexed to some subfunction  $\mathbf{h}_i$ :

$$\mathbf{X} \stackrel{\text{def}}{=} \{x \in \{0, 1\}^n : |x|_M \in [n/4 - \sqrt{n}, n/4 + \sqrt{n}] \text{ and } \Gamma_{\mathbf{T}}(x) \in [N]\},$$

and define  $\gamma \in (0, 1)$  be the parameter:  $\gamma \stackrel{\text{def}}{=} \mathbb{E}_{\mathbf{T} \sim \mathcal{E}(M')} \left[ \frac{|\mathbf{X}|}{2^n} \right]$ .

In addition, let  $X_i \subseteq X$  be the subset of points  $x \in X$  with  $\Gamma_{\mathbf{T}}(x) = i$ , and note that the subsets  $X_1, \dots, X_N$  partition  $X$ , where each  $|X_i| \leq 2^{n-\sqrt{n}}$ . With probability  $1 - o(1)$  over the draw of  $\mathbf{T} \sim \mathcal{E}(M)$ , we have:

$$\sum_{i=1}^{3N/4} |X_i| = 2^n \cdot \frac{3\gamma}{4} \left( 1 \pm \frac{1}{n} \right) \quad \text{and} \quad \sum_{i=3N/4+1}^N |X_i| = 2^n \cdot \frac{\gamma}{4} \left( 1 \pm \frac{1}{n} \right). \quad (3)$$

Thus, we only consider functions  $\mathbf{f} \sim \mathcal{D}_{\text{yes}}$  (or  $\sim \mathcal{D}_{\text{no}}$ ) where the sets  $M$ , and  $T$  satisfy (3).

We consider any set  $A \subseteq \overline{M}$  of size  $\frac{n}{4}$ . Now, consider any graph  $G$  defined over vertices in  $\overline{M}$ , and we let:

$$\chi(G) = \min \left\{ \frac{E_G(S, S) + E_G(\overline{S}, \overline{S})}{E_G(\overline{M}, \overline{M})} : S \subseteq \overline{M} \right\}.$$

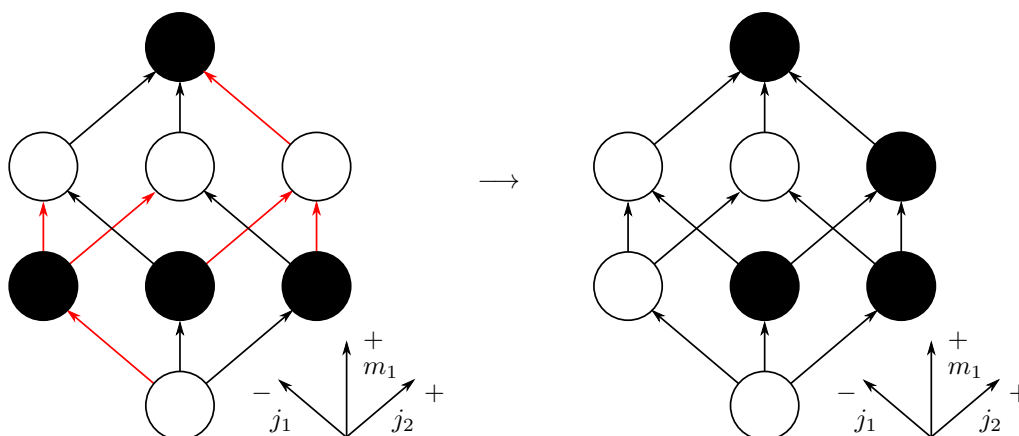
In other words, we note that  $\chi(G)$  is one minus the fractional value of the maximum cut, and the value of  $\chi(G)$  is minimized for the set  $S$  achieving the maximum cut of  $G$ . The following lemma relates the distance to unateness of a function  $\mathbf{f} = f_{T, A, \mathbf{H}}$  with  $\mathbf{H} \sim \mathcal{H}(m_1, m_2, G)$ , where  $G$  is an underlying graph defined on vertices in  $\overline{M}$ .

► **Lemma 10.** *Let  $G$  be any graph defined over vertices in  $\overline{M}$ . If  $\mathbf{f} = f_{T, A, \mathbf{H}}$  where  $\mathbf{H} \sim \mathcal{H}(m_1, m_2, G)$ , then with probability at least  $1 - o(1)$ ,*

$$\frac{\gamma}{16} \left( 1 + \frac{1}{2} \cdot \chi(G) \right) - o(1) \leq \text{dist}(\mathbf{f}, \text{Unate}) \leq \frac{\gamma}{16} \left( 1 + \frac{1}{2} \cdot \chi(G) \right) + o(1).$$

We consider the constants  $\varepsilon_0 = \frac{\gamma}{16}$  and  $\varepsilon_1 = \frac{5\gamma}{64}$ .

► **Corollary 11.** *We have that  $\mathbf{f} \sim \mathcal{D}_{\text{yes}}$  has  $\text{dist}(\mathbf{f}, \text{Unate}) \leq \varepsilon_0 + o(1)$  with high probability, and  $\mathbf{f} \sim \mathcal{D}_{\text{no}}$  has  $\text{dist}(\mathbf{f}, \text{Unate}) \geq \varepsilon_1 - o(1)$  with high probability.*



■ **Figure 4** Similarly to Figure 2, this is an example of a function  $h_i: \{0,1\}^n \rightarrow \{0,1\}$  with  $h_i(x) = x_{j_1} \oplus x_{j_2} \oplus x_{m_1}$  variables  $j_1$  (which ought to be anti-monotone),  $j_2$  (which ought to be monotone), and  $m_1$  (which is always monotone) being “fixed” into a function  $h'_i: \{0,1\}^n \rightarrow \{0,1\}$  defined on the right-hand side.

### 4.3 Reducing from Rejection Sampling

In order to reduce from rejection sampling, we need the following two lemmas.

► **Lemma 12.** *Suppose there exists a deterministic algorithm Alg making  $q$  queries to Boolean functions  $f: \{0,1\}^{2n} \rightarrow \{0,1\}$ . Then, there exists a deterministic non-adaptive algorithm Alg' making rejection sampling queries to an  $n$ -vertex graph with  $\text{cost}(\text{Alg}') = qn$  such that:*

$$\begin{aligned} \Pr_{f \sim \mathcal{D}_{\text{yes}}} [\text{Alg}(f) \text{ “accepts”}] &= \Pr_{\mathbf{G} \sim \mathcal{G}_2} [\text{Alg}'(\mathbf{G}) \text{ outputs “}\mathcal{G}_2\text{”}], \quad \text{and} \\ \Pr_{f \sim \mathcal{D}_{\text{no}}} [\text{Alg}(f) \text{ “accepts”}] &= \Pr_{\mathbf{G} \sim \mathcal{G}_1} [\text{Alg}'(\mathbf{G}) \text{ outputs “}\mathcal{G}_2\text{”}]. \end{aligned}$$

► **Lemma 13.** *Suppose there exists a deterministic non-adaptive algorithm Alg making  $q$  queries to Boolean functions  $f: \{0,1\}^{2n} \rightarrow \{0,1\}$  where  $q \leq \frac{n^{3/2}}{\log^8 n}$ . Then, there exists a deterministic non-adaptive algorithm Alg' making rejection sampling queries to an  $n$ -vertex graph such that:*

$$\begin{aligned} \Pr_{f \sim \mathcal{D}_{\text{yes}}} [\text{Alg}(f) \text{ “accepts”}] &\approx \Pr_{\mathbf{G} \sim \mathcal{G}_2} [\text{Alg}'(\mathbf{G}) \text{ outputs “}\mathcal{G}_2\text{”}] \pm o(1), \quad \text{and} \\ \Pr_{f \sim \mathcal{D}_{\text{no}}} [\text{Alg}(f) \text{ “accepts”}] &\approx \Pr_{\mathbf{G} \sim \mathcal{G}_1} [\text{Alg}'(\mathbf{G}) \text{ outputs “}\mathcal{G}_2\text{”}] \pm o(1). \end{aligned}$$

and has  $\text{cost}(\text{Alg}') \leq q\sqrt{n} \log n$  with probability  $1 - o(1)$  over the randomness in Alg'.

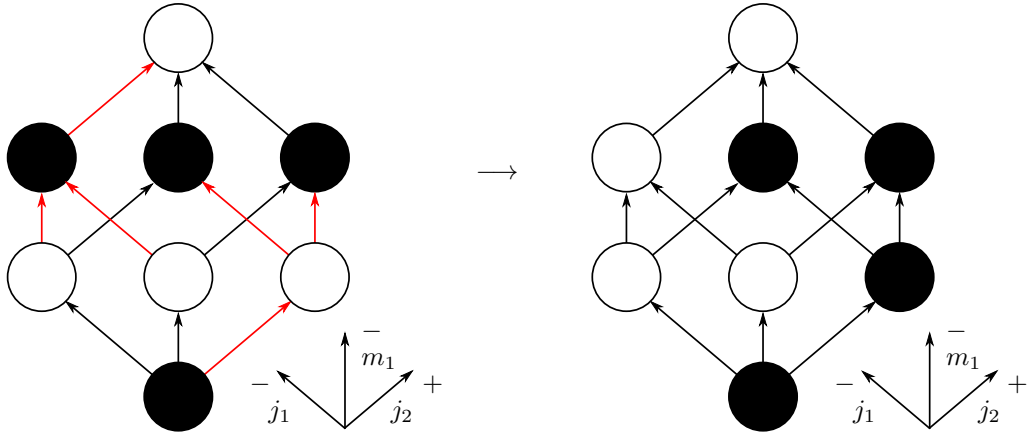
Combining Lemma 12 with Theorem 1, we conclude Theorem 4, and combining Lemma 13 with Theorem 1, we conclude Theorem 5.

## 5 A lower bound for distinguishing $\mathcal{G}_1$ and $\mathcal{G}_2$ with rejection samples

In this section, we derive a lower bound for distinguishing  $\mathcal{G}_1$  and  $\mathcal{G}_2$  with rejection samples.

► **Lemma 14.** *Any deterministic non-adaptive algorithm Alg with  $\text{cost}(\text{Alg}) \leq \frac{n^2}{\log^6 n}$ , has:*

$$\Pr_{\mathbf{G} \sim \mathcal{G}_1} [\text{Alg outputs “}\mathcal{G}_1\text{”}] \leq (1 + o(1)) \Pr_{\mathbf{G} \sim \mathcal{G}_2} [\text{Alg outputs “}\mathcal{G}_1\text{”}] + o(1).$$



■ **Figure 5** Similarly to Figure 2, this is an example of a function  $h_i: \{0, 1\}^n \rightarrow \{0, 1\}$  with  $h_i(x) = \neg x_{j_1} \oplus x_{j_2} \oplus x_{m_2}$  variables  $j_1$  (which ought to be anti-monotone),  $j_2$  (which ought to be monotone), and  $m_2$  (which is always anti-monotone) being “fixed” into a function  $h'_i: \{0, 1\}^n \rightarrow \{0, 1\}$  defined on the right-hand side.

We assume Alg is a deterministic non-adaptive algorithm with  $\text{cost}(\text{Alg}) \leq \frac{n^2}{\log^6 n}$ . Alg makes queries  $L_1, \dots, L_t \subseteq [n]$  and the oracle returns  $\mathbf{v}_1, \dots, \mathbf{v}_t$ , some of which are edges, some are lone vertices, and some are  $\emptyset$ . Let  $\mathbf{G}_o \subseteq \mathbf{G}$  be the graph observed by the algorithm by considering all edges in  $\mathbf{v}_1, \dots, \mathbf{v}_t$ . We let  $|\mathbf{G}_o|$  be the number of edges.

Before going on to prove the lower bound, we use the following simplification. First, we assume that any algorithm Alg has all its queries  $L_1, \dots, L_t$  satisfying that either  $|L_i| \leq \frac{n}{\log n}$ , or  $L_i = [n]$ . Thus, it suffices to show for this restricted class of algorithms, the cost must be at least  $\frac{n^2}{\log^5 n}$ .

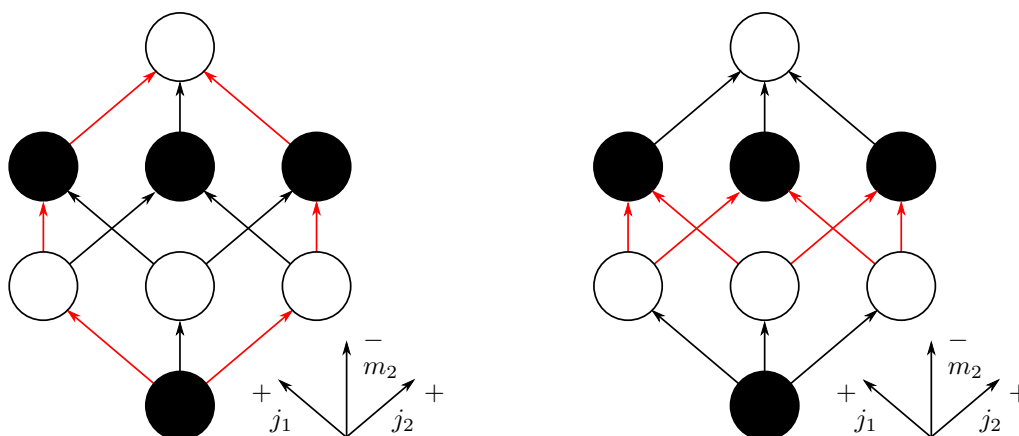
### 5.1 High Level Overview

We will argue outcome-by-outcome; i.e., we consider the possible ways the algorithm can act, which depends on the responses to the queries the algorithm gets. Consider some responses  $v_1, \dots, v_t \in [n] \cup ([n] \times [n]) \cup \{\emptyset\}$ , where each  $v_i$  may be either a lone vertex, an edge, or  $\emptyset$ . Suppose that upon observing this outcome, the algorithm outputs “ $\mathcal{G}_1$ ”. There will be two cases:

- The first case is when the probability of observing this outcome from  $\mathcal{G}_2$  is not too much lower than the probability of observing this outcome from  $\mathcal{G}_1$ . In these outcomes, we will not get too much advantage in distinguishing  $\mathcal{G}_1$  and  $\mathcal{G}_2$ .
- The other case is when the probability of observing this outcome from  $\mathcal{G}_2$  is substantially lower than the probability of observing this outcome from  $\mathcal{G}_1$ . These cases do help us distinguish between  $\mathcal{G}_1$  and  $\mathcal{G}_2$ ; thus, we will want to show that collectively, the probability that we observe these outcomes from  $\mathcal{G}_1$  is  $o(1)$ .

We will be able to characterize the outcomes which fall into the first case and the second case by considering a sequence of events. In particular we define five events which depend on  $v_1, \dots, v_t$ , as well as the random choice of  $\mathbf{A}$ . Consider the outcome  $v_1, \dots, v_t$  which together form components  $C_1, \dots, C_\alpha$ . The events are the following<sup>7</sup>:

<sup>7</sup> We note that the first two event are not random and depends on the values  $v_1, \dots, v_t$ , and the rest are random variables depending on the partition  $\mathbf{A}$  and the oracle responses  $v_1, \dots, v_t$ .



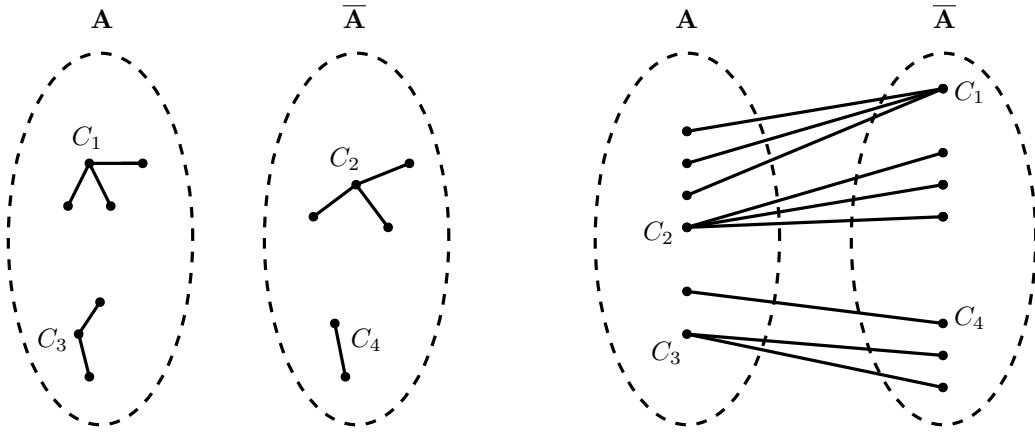
■ **Figure 6** Examples of functions  $h_i: \{0, 1\}^n \rightarrow \{0, 1\}$  with orientations on the variables and violating edges. On the left-hand side,  $h_i(x) = \neg x_{j_1} \oplus x_{j_2} \oplus x_{m_2}$  with variables  $j_1$  and  $j_2$  (which ought to be monotone), and  $m_2$  (which is always anti-monotone). On the right-hand side,  $h_i(x) = \neg x_{j_1} \oplus x_{j_2} \oplus x_{m_2}$  with variables  $j_1$  and  $j_2$  (which ought to be anti-monotone), and  $m_2$  (which is always anti-monotone). We note that the violating edges form a cycle of length 6, so any unate function whose orientations on  $j_1$  and  $j_2$  are as indicated (both monotone on the left-hand side, and both anti-monotone on the right-hand side) must disagree on a  $\frac{3}{8}$ -fraction of the points.

1.  $\mathcal{E}_T$  (Observe small trees): this is the event where the values of  $v_1, \dots, v_t$  form components  $C_1, \dots, C_\alpha$  which are all trees of size at most  $\log n$ .
2.  $\mathcal{E}_F$  (Observe few non-empty responses): this is the event where the values of  $v_1, \dots, v_t$  have at most  $\frac{n}{\log^4 n}$  non- $\emptyset$  responses. This event implies that the total number of vertices in the responses  $v_1, \dots, v_t$  is at most  $\frac{n}{\log^4 n}$ .
3.  $\mathcal{E}_{C,\text{yes}}$  and  $\mathcal{E}_{C,\text{no}}$  (Consistency condition of the components observed): these are the events where  $\mathbf{A} \subseteq [n]$  partitions the components  $C_1, \dots, C_\alpha$  in a manner consistent with  $\mathcal{G}_1$  in  $\mathcal{E}_{C,\text{yes}}$  or  $\mathcal{G}_2$  in  $\mathcal{E}_{C,\text{no}}$ , i.e., either every  $C_i$  is contained within  $\mathbf{A}$  or  $\overline{\mathbf{A}}$  (in the case of  $\mathcal{G}_1$ , or edges in every  $C_i$  cross the partition on vertices induced by  $\mathbf{A}$  (in the case of  $\mathcal{G}_2$ ). These events are random variables that depend only on  $\mathbf{A}$ . It will become clear that in order to observe the outcome  $v_1, \dots, v_t$  in  $\mathcal{G}_1$ , event  $\mathcal{E}_{C,\text{yes}}$  must be triggered, and in  $\mathcal{G}_2$ , event  $\mathcal{E}_{C,\text{no}}$  must be triggered. See Figure 7 for an illustration.
4.  $\mathcal{E}_O$  (Observe specific responses): this event is over the randomness in  $\mathbf{A}$ , as well as the randomness in the responses of the oracle  $v_1, \dots, v_t$ . The event is triggered when the responses of the oracle are exactly those dictated by  $v_1, \dots, v_t$ ; i.e., for all  $i \in [t]$ ,  $v_i = v_i$ .
5.  $\mathcal{E}_B$  (Balanced lone vertices condition): this event is over the randomness in  $\mathbf{A}$ , as well as the responses  $v_1, \dots, v_t$ . The event occurs when the queries  $L_i$  corresponding to lone vertices  $v_i$  have  $|L_i \cap \mathbf{A}|$  and  $|L_i \cap \overline{\mathbf{A}}|$  roughly equal, and roughly half of  $v_i$  fall in  $\mathbf{A}$ .

Having defined these events, the lower bound follows by the following three lemmas. The first lemma says that for any outcomes satisfying  $\mathcal{E}_T$  and  $\mathcal{E}_F$ , the probability over  $\mathbf{A}$  of being consistent in  $\mathcal{G}_1$  cannot be much higher than in  $\mathcal{G}_2$ . The second lemma says that the outcomes satisfying the events described above do not help in distinguishing  $\mathcal{G}_1$  and  $\mathcal{G}_2$ . The third lemma says that good outcomes occur with high probability over  $\mathcal{G}_1$ .

► **Lemma 15 (Consistency Lemma).** *Consider a fixed  $v_1, \dots, v_t \in [n] \cup ([n] \times [n]) \cup \{\emptyset\}$  forming components  $C_1, \dots, C_\alpha$  where events  $\mathcal{E}_T$  and  $\mathcal{E}_F$  are satisfied. Then, we have:*

$$\Pr_{\substack{\mathbf{G} \sim \mathcal{G}_1 \\ v_1, \dots, v_t}} [\mathcal{E}_{C,\text{yes}}] \leq (1 + o(1)) \Pr_{\substack{\mathbf{G} \sim \mathcal{G}_2 \\ v_1, \dots, v_t}} [\mathcal{E}_{C,\text{no}}].$$



■ **Figure 7**  $A$  consistently partition of the components  $C_1, C_2, C_3$  and  $C_4$  according to  $\mathcal{G}_1$  (on the left) and  $\mathcal{G}_2$  (on the right).

► **Lemma 16** (Good Outcomes Lemma). Consider a fixed  $v_1, \dots, v_t \in [n] \cup ([n] \times [n]) \cup \{\emptyset\}$  forming components  $C_1, \dots, C_\alpha$  where events  $\mathcal{E}_T$  and  $\mathcal{E}_F$  are satisfied. Then, we have:

$$\Pr_{\substack{\mathbf{G} \sim \mathcal{G}_1 \\ v_1, \dots, v_t}} [\mathcal{E}_O \wedge \mathcal{E}_B \mid \mathcal{E}_{C, \text{yes}}] \leq (1 + o(1)) \Pr_{\substack{\mathbf{G} \sim \mathcal{G}_2 \\ v_1, \dots, v_t}} [\mathcal{E}_O \mid \mathcal{E}_{C, \text{no}}].$$

► **Lemma 17** (Bad Outcomes Lemma). We have that:

$$\Pr_{\substack{\mathbf{G} \sim \mathcal{G}_1 \\ v_1, \dots, v_t}} [\neg \mathcal{E}_T \vee \neg \mathcal{E}_F \vee \neg \mathcal{E}_B] = o(1).$$

Assuming the above three lemmas, we may prove Lemma 14.

## References

- 1 Nir Ailon, Bernard Chazelle, Seshadhri Comandur, and Ding Liu. Estimating the distance to a monotone function. *Random Structures and Algorithms*, 31(3):371–383, 2007.
- 2 Rokhsana Baleshzar, Deeparnab Chakrabarty, Ramesh Krishnan S. Pallavoor, Sofya Raskhodnikova, and C. Seshadhri. A lower bound for nonadaptive, one-sided error testing of unateness of Boolean functions over the hypercube. *arXiv preprint*, 2017. arXiv:1706.00053.
- 3 Rokhsana Baleshzar, Deeparnab Chakrabarty, Ramesh Krishnan S. Pallavoor, Sofya Raskhodnikova, and C. Seshadhri. Optimal Unateness Testers for Real-Values Functions: Adaptivity Helps. In *Proceedings of the 44th International Colloquium on Automata, Languages and Programming (ICALP '2017)*, 2017.
- 4 Aleksandrs Belovs and Eric Blais. A polynomial lower bound for testing monotonicity. In *Proceedings of the 48th ACM Symposium on the Theory of Computing (STOC '2016)*, pages 1021–1032, 2016.
- 5 Piotr Berman, Meiram Murzabulatov, and Sofya Raskhodnikova. Tolerant Testers of Image Properties. In *Proceedings of the 43th International Colloquium on Automata, Languages and Programming (ICALP '2016)*, pages 90:1–90:14, 2016.
- 6 Piotr Berman, Sofya Raskhodnikova, and Grigory Yaroslavtsev.  $L_p$ -testing. In *Proceedings of the 46th ACM Symposium on the Theory of Computing (STOC '2014)*, 2014.
- 7 Eric Blais. Improved bounds for testing juntas. In *Approximation, Randomization and Combinatorial Optimization. Algorithms and Techniques*, pages 317–330. Springer, 2008.



- 8 Eric Blais. Testing juntas nearly optimally. In *Proceedings of the 41st ACM Symposium on the Theory of Computing (STOC '2009)*, pages 151–158, 2009.
- 9 Eric Blais, Clément L Canonne, Talya Eden, Amit Levi, and Dana Ron. Tolerant junta testing and the connection to submodular optimization and function isomorphism. In *Proceedings of the 29th ACM-SIAM Symposium on Discrete Algorithms (SODA '2018)*, pages 2113–2132, 2018.
- 10 Harry Buhrman, David Garcia-Soriano, Arie Matsliah, and Ronald de Wolf. The non-adaptive query complexity of testing  $k$ -parities. *Chicago Journal of Theoretical Computer Science*, 6:1–11, 2013.
- 11 Andrea Campagna, Alan Guo, and Ronitt Rubinfeld. Local reconstructors and tolerant testers for connectivity and diameter. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*, pages 411–424. Springer, 2013.
- 12 Clément L. Canonne, Dana Ron, and Rocco A. Servedio. Testing probability distributions using conditional samples. *SIAM Journal on Computing*, 44(3):540–616, 2015.
- 13 Deeparnab Chakrabarty and Seshadhri Comandur. An  $o(n)$  monotonicity tester for boolean functions over the hypercube. *SIAM Journal on Computing*, 45(2):461–472, 2016.
- 14 Deeparnab Chakrabarty and C. Seshadhri. A  $\tilde{O}(n)$  non-adaptive tester for unateness. *arXiv preprint*, 2016. [arXiv:1608.06980](https://arxiv.org/abs/1608.06980).
- 15 Sourav Chakraborty, Eldar Fischer, David García-Soriano, and Arie Matsliah. Junta-symmetric functions, hypergraph isomorphism and crunching. In *Proceedings of the 27th Conference on Computational Complexity (CCC '2012)*, pages 148–158. IEEE, 2012.
- 16 Sourav Chakraborty, Eldar Fischer, Yonatan Goldhirsh, and Arie Matsliah. On the power of conditional samples in distribution testing. *SIAM Journal on Computing*, 45(4):1261–1296, 2016.
- 17 Xi Chen, Rocco A. Servedio, Li-Yang Tan, Erik Waingarten, and Jinyu Xie. Settling the query complexity of non-adaptive junta testing. In *Proceedings of the 32nd Conference on Computational Complexity (CCC '2017)*, 2017.
- 18 Xi Chen, Erik Waingarten, and Jinyu Xie. Beyond Talagrand functions: new lower bounds for testing monotonicity and unateness. In *Proceedings of the 49th ACM Symposium on the Theory of Computing (STOC '2017)*, 2017.
- 19 Xi Chen, Erik Waingarten, and Jinyu Xie. Boolean Unateness Testing with  $\tilde{O}(n^{3/4})$  Adaptive Queries. In *Proceedings of the 58th Annual IEEE Symposium on Foundations of Computer Science (FOCS '2017)*, 2017.
- 20 Hana Chockler and Dan Gutfreund. A lower bound for testing juntas. *Information Processing Letters*, pages 301–305, 2004.
- 21 Ilias Diakonikolas, Homin K Lee, Kevin Matulef, Krzysztof Onak, Ronitt Rubinfeld, Rocco A Servedio, and Andrew Wan. Testing for concise representations. In *Proceedings of the 48th Annual IEEE Symposium on Foundations of Computer Science (FOCS '2007)*, pages 549–558. IEEE, 2007.
- 22 Shahar Fattal and Dana Ron. Approximating the distance to monotonicity in high dimensions. *ACM Transactions on Algorithms*, 6(3):52, 2010.
- 23 Eldar Fischer and Lance Fortnow. Tolerant versus intolerant testing for Boolean properties. *Theory of Computing*, 2(9):173–183, 2006.
- 24 Eldar Fischer, Guy Kindler, Dana Ron, Shmuel Safra, and Alex Samorodnitsky. Testing juntas. *Journal of Computer and System Sciences*, 68(4):753–787, 2004. doi:10.1016/j.jcss.2003.11.004.
- 25 Eldar Fischer and Ilan Newman. Testing versus estimation of graph properties. *SIAM Journal on Computing*, 37(2):482–501, 2007.
- 26 Oded Goldreich. *Introduction to property testing*. Cambridge University Press, 2017.

- 27 Oded Goldreich, Shafi Goldwasser, Eric Lehman, Dana Ron, and Alex Samordinsky. Testing Monotonicity. *Combinatorica*, 20(3):301–337, 2000.
- 28 Oded Goldreich, Shafi Goldwasser, and Dana Ron. Property testing and its connection to learning and approximation. *Journal of the ACM*, 45(4):653–750, 1998.
- 29 Oded Goldreich and Dana Ron. On sample-based testers. *ACM Transactions on Computation Theory*, 8(2), 2016.
- 30 Venkatesan Guruswami and Atri Rudra. Tolerant locally testable codes. In *Approximation, Randomization and Combinatorial Optimization. Algorithms and Techniques*, pages 306–317. Springer, 2005.
- 31 Subhash Khot and Igor Shinkar. An  $\tilde{O}(n)$  queries adaptive tester for unateness. In *Approximation, Randomization and Combinatorial Optimization. Algorithms and Techniques*, pages 37:1–37:7, 2016.
- 32 Swastik Kopparty and Shubhangi Saraf. Tolerant linearity testing and locally testable codes. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*, pages 601–614. Springer, 2009.
- 33 Sharon Marko and Dana Ron. Approximating the distance to properties in bounded-degree and general sparse graphs. *ACM Transactions on Algorithms*, 5(2):22, 2009.
- 34 Michal Parnas, Dana Ron, and Ronitt Rubinfeld. Tolerant property testing and distance approximation. *Journal of Computer and System Sciences*, 72(6):1012–1042, 2006.
- 35 Dana Ron. Property testing: A learning theory perspective. *Foundations and Trends® in Machine Learning*, 1(3):307–402, 2008.
- 36 Dana Ron. Algorithmic and analysis techniques in property testing. *Foundations and Trends® in Theoretical Computer Science*, 5(2):73–205, 2010.
- 37 Rocco A Servedio, Li-Yang Tan, and John Wright. Adaptivity helps for testing juntas. In *Proceedings of the 30th Conference on Computational Complexity (CCC '2015)*, pages 264–279, 2015.
- 38 Roei Tell. A Note on Tolerant Testing with One-Sided Error. In *Electronic Colloquium on Computational Complexity (ECCC)*, volume 23, page 32, 2016.

# Secret Sharing with Binary Shares

**Fuchun Lin**

Division of Mathematical Sciences, School of Physical and Mathematical Sciences,  
Nanyang Technological University, SG  
felin@ntu.edu.sg

**Mahdi Cheraghchi**

Department of Computing, Imperial College London, UK  
m.cheraghchi@imperial.ac.uk

**Venkatesan Guruswami**

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

**Reihaneh Safavi-Naini**

Department of Computer Science, University of Calgary, CA  
rei@ucalgary.ca

**Huaxiong Wang**

Division of Mathematical Sciences, School of Physical and Mathematical Sciences,  
Nanyang Technological University, SG  
hxwang@ntu.edu.sg

---

## Abstract

Shamir's celebrated secret sharing scheme provides an efficient method for encoding a secret of arbitrary length  $\ell$  among any  $N \leq 2^\ell$  players such that for a threshold parameter  $t$ , (i) the knowledge of any  $t$  shares does not reveal any information about the secret and, (ii) any choice of  $t + 1$  shares fully reveals the secret. It is known that any such threshold secret sharing scheme necessarily requires shares of length  $\ell$ , and in this sense Shamir's scheme is optimal. The more general notion of ramp schemes requires the reconstruction of secret from any  $t + g$  shares, for a positive integer gap parameter  $g$ . Ramp secret sharing scheme necessarily requires shares of length  $\ell/g$ . Other than the bound related to secret length  $\ell$ , the share lengths of ramp schemes can not go below a quantity that depends only on the gap ratio  $g/N$ .

In this work, we study secret sharing in the extremal case of bit-long shares and arbitrarily small gap ratio  $g/N$ , where standard ramp secret sharing becomes impossible. We show, however, that a slightly relaxed but equally effective notion of semantic security for the secret, and negligible reconstruction error probability, eliminate the impossibility. Moreover, we provide explicit constructions of such schemes. One of the consequences of our relaxation is that, unlike standard ramp schemes with perfect secrecy, adaptive and non-adaptive adversaries need different analysis and construction. For non-adaptive adversaries, we explicitly construct secret sharing schemes that provide secrecy against any  $\tau$  fraction of observed shares, and reconstruction from any  $\rho$  fraction of shares, for any choices of  $0 \leq \tau < \rho \leq 1$ . Our construction achieves secret length  $N(\rho - \tau - o(1))$ , which we show to be optimal. For adaptive adversaries, we construct explicit schemes attaining a secret length  $\Omega(N(\rho - \tau))$ . We discuss our results and open questions.

**2012 ACM Subject Classification** Security and privacy → Information-theoretic techniques, Theory of computation → Expander graphs and randomness extractors, Theory of computation → Error-correcting codes

**Keywords and phrases** Secret sharing scheme, Wiretap channel

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2019.53



© Fuchun Lin, Mahdi Cheraghchi, Venkatesan Guruswami, Reihaneh Safavi-Naini, and Huaxiong Wang;

licensed under Creative Commons License CC-BY

10th Innovations in Theoretical Computer Science (ITCS 2019).

Editor: Avrim Blum; Article No. 53; pp. 53:1–53:20



Leibniz International Proceedings in Informatics

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

**Related Version** Full version at <https://eprint.iacr.org/2018/746>.

**Funding** The research of Fuchun Lin and Huaxiong Wang was supported by Singapore Ministry of Education under Research Grant MOE2016-T2-2-014(S) and RG133/17(S). The research of Venkatesan Guruswami was supported in part by United States NSF grants CCF-1422045 and CCF-1563742. The research of Reihaneh Safavi-Naini was in part supported by Natural Sciences and Engineering Research Council of Canada, Discovery Grants Program.

**Acknowledgements** We would like to thank the anonymous reviewers for comments that improved the presentation of the paper.

## 1 Introduction

Secret sharing, introduced independently by Blakley [3] and Shamir [20], is one of the most fundamental cryptographic primitives with far-reaching applications, such as being a major tool in secure multiparty computation (cf. [12]). The general goal in secret sharing is to encode a secret  $s$  into a number of *shares*  $X_1, \dots, X_N$  that are distributed among  $N$  players such that only certain *authorized subsets* of the players can reconstruct the secret. An *authorized* subset of players is a set  $A \subseteq [N]$  such that the set of shares with indices in  $A$  can collectively be used to reconstruct the secret  $s$  (*perfect reconstructibility*). On the other hand,  $A$  is an *unauthorized* subset if the knowledge of the shares with indices in  $A$  reveals no information about the secret (*perfect privacy*). The set of authorized and unauthorized sets define an access structure, of which the most widely used is the so-called *threshold structure*. A secret sharing scheme with *threshold* access structure, is defined with respect to an integer parameter  $t$  and satisfies the following properties. Any set  $A \subseteq [N]$  with  $|A| \leq t$  is an unauthorized set. That is, the knowledge of any  $t$  shares, or fewer, does not reveal any information about the secret. On the other hand, any set  $A$  with  $|A| > t$  is an authorized set. That is, the knowledge of any  $t + 1$  or more shares completely reveals the secret.

Shamir's secret sharing scheme [20] gives an elegant construction for the threshold access structure that can be interpreted as the use of Reed-Solomon codes for encoding the secret. Suppose the secret  $s$  is an  $\ell$ -bit string and  $N \leq 2^\ell$ . Then, Shamir's scheme treats the secret as an element of the finite field  $\mathbb{F}_q$ , where  $q = 2^\ell$ , padded with  $t$  uniformly random and independent elements from the same field. The resulting vector over  $\mathbb{F}_q^{t+1}$  is then encoded using a Reed-Solomon code of length  $N$ , providing  $N$  shares of length  $\ell$  bits each. The fact that a Reed-Solomon code is Maximum Distance Separable (MDS) can then be used to show that the threshold guarantee for privacy and reconstruction is satisfied.

Remarkably, Shamir's scheme is optimal for threshold secret sharing in the following sense: Any threshold secret sharing scheme sharing  $\ell$ -bit secrets necessarily requires shares of length at least  $\ell$ , and Shamir's scheme attains this lower bound [23]. It is natural to ask whether secret sharing is possible at share lengths below the secret length  $\log q < \ell$ , where  $\log$  is to base 2 throughout this work. Of course, in this case, the threshold guarantee that requires all subsets of participants be either authorized, or unauthorized, can no longer be attained. Instead, the notion can be relaxed to *ramp* secret sharing which allows some subset of participants to learn some information about the secret. A ramp scheme is defined with respect to two threshold parameters,  $t$  and  $r > t + 1$ . As in threshold scheme, the knowledge of any  $t$  shares or fewer does not reveal any information about the secret. On the other hand, any  $r$  shares can be used to reconstruct the secret. The subsets of size between  $t + 1$  and  $r - 1$ , may learn some information about the secret. The information-theoretic bound (see e.g. [17]) now becomes

$$\ell \leq (r - t) \log q. \tag{1}$$

Ideally, one would like to obtain equality in (1) for as general parameter settings as possible.

Let  $g := r - t$  denote the *gap* between the privacy and reconstructibility parameters. Let the secret length  $\ell$  and the number of players  $N$  be unconstrained integer parameters. It is known that, using Reed-Solomon code interpretation of Shamir's approach applied to a random linear code, for every fixed *relative gap*  $\gamma := g/N$ , there is a constant  $q$  only depending on  $\gamma$  such that a ramp secret sharing scheme with share size  $q$  exists. Such schemes can actually be constructed by using explicit algebraic geometry codes instead of random linear codes. In fact, this dependence of share size  $q$  on relative gap  $g/N$  is inherent for threshold and more generally ramp schemes. It is shown in an unpublished work of Kilian and Nisan<sup>1</sup> for threshold schemes, and later more generally in [8], that for ramp schemes with share size  $q$ , threshold gap  $g$ , privacy threshold  $t$  and unconstrained number of players  $N$ , the following always holds:  $q \geq (N - t + 1)/g$ . Very recently in [4], a new bound with respect to the reconstruction parameter  $r$  is proved through connecting secret sharing for one bit secret to game theory:  $q \geq (r + 1)/g$ . These two bounds together yield

$$q \geq (N + g + 2)/(2g). \quad (2)$$

Note that the bound (2) is very different from the bound (1) in nature. The bound (1) is the fundamental limitation of information-theoretic security, bearing the same flavour as the One-Time-Pad. The bound (2) is independent of the secret length and holds even when the secret is one bit.

We ask the following question: *For a fixed constant share size  $q$  (in particular,  $q = 2$ ), is it possible to construct (relaxed but equally effective) ramp secret sharing schemes with arbitrarily small relative gap  $\gamma > 0$  that asymptotically achieve equality in (1)?*

Our results in this work show that the restriction (2) can be overcome if we allow a negligible privacy error in statistical distance (semantic security) and a negligible reconstruction error probability.

## 1.1 Related work

Secret sharing with binary shares is recently selectively studied in [5] with the focus on the tradeoff between the privacy parameter  $t$  and the computational complexity of the reconstruction function. The concern of this line of works is to construct secret sharing schemes whose sharing and reconstructing algorithms are both in the complexity class  $AC^0$  (i.e., constant depth circuits).

The model of secret sharing considered in [5] has perfect privacy, namely, the distributions of any  $t$  shares from a pair of distinct secrets are identical, while reconstruction is not necessarily with probability 1. In a followup work [6], the privacy is further relaxed to semantic security with an error parameter  $\varepsilon > 0$ . The relaxation is shown to be useful in allowing more choices of  $t$  while keeping the computational complexity of the reconstruction algorithm within  $AC^0$ .

In [5, 6], the secret is one bit. In [9], secrets of length equal to a fraction of  $N$  (number of players) is considered. This time binary secret sharing with adaptive and non-adaptive adversaries similar to the model we consider in this work is defined. However the paper considers only a privacy threshold  $t$ , and reconstruction is from the full share set ( $r = N$  always). Their goal is to achieve large secrets  $\ell = \Omega(N)$  over binary shares with large privacy parameter  $t = \Omega(N)$ , which is also similar to ours. They have an additional goal of keeping

<sup>1</sup> Their result is unpublished, and is independently obtained and generalised by [8] (see [8, Appendix A]).

the computational complexity of the reconstruction algorithm within  $AC^0$ , which we do not consider in this work. Their large privacy parameter  $t = \tau N$  is with a  $\tau$  much smaller than 1, which means that the relative threshold gap  $\gamma = 1 - \tau$  can not be arbitrarily small.

In the literature, perhaps the closest notion to secret sharing with binary shares is that of *wiretap codes*, first studied in information theory. In the basic wiretap channel model of Wyner and its extension to *broadcast channel with confidential messages* [25, 13], there is a point-to-point *main* channel between a sender and a receiver that has partial leakage to the adversary, and the leakage of information is modelled by a second point-to-point *wiretapper* channel between the sender and the adversary. The goal of the sender is to encode messages in such a way that the receiver can decode them, while the adversary does not learn much about them [25]. The highest information rate achievable for a wiretap channel is called the *secrecy capacity*. A Binary Erasure Channel with erasure probability  $p$  ( $BEC_p$ ) is a probabilistic transformation that maps inputs 0 and 1 to a symbol  $?$ , which denotes erasure, with probability  $p$  and to the inputs themselves with probability  $1 - p$ . The erasure channel scenario of wiretap model is closely related to secret sharing with fixed share size. For a pair ( $BEC_{p_m}, BEC_{p_w}$ ) of  $BEC$ 's, such that  $p_m < p_w$ , it is known that the secrecy capacity is the difference of the respective channel capacities:  $(1 - p_m) - (1 - p_w) = p_w - p_m$ .

It is important to note the distinctions between the erasure wiretap model above, and binary secret sharing. First, the guarantees of wiretap codes are required to only hold for random messages, whereas in secret sharing, the cryptographic convention of security for worst-case messages is required. Second, in the standard wiretap model, the notion of secrecy is information-theoretic and is typically measured in terms of the mutual information. Namely, for a random  $\ell$ -bit message  $M$ , and letting  $W$  denote the information delivered to the adversary, secrecy is satisfied in the weak (resp., strong) sense if  $I(M; W) \leq \varepsilon \ell$  (resp.,  $I(M; W) \leq \varepsilon$ ) for an arbitrarily small constant  $\varepsilon$ . The randomness of the random variable  $W$  depends on the randomness of  $M$ , the two channels, as well as the internal randomness of the encoder. For secret sharing, on the other hand, either perfect secrecy or semantic secrecy (negligible leakage with respect to statistical distance) is a requirement.

The notion of secrecy in wiretap codes has evolved over years. More recently the notion of semantic security for wiretap model has been introduced [2], which allows arbitrary message distribution and is shown to be equivalent to negligible leakage with respect to statistical distance.

There remains one last distinction between *semantically secure* wiretap model and secret sharing with fixed share size. That is the nature of the main and wiretapper channels are typically stochastic (e.g., the erasure channel with random i.i.d. erasures), whereas for secret sharing a worst-case guarantee for the erasure patterns is required. Namely, in secret sharing, reconstruction with overwhelming probability is required for every choice of  $r$  or more shares, and privacy of the secret is required for every (adaptive or non-adaptive) choice of the  $t$  shares observed by the adversary.

On the other hand, Ozarow and Wyner proposed the *wiretap channel II*, where an adversary observes arbitrary  $t$  out of the total  $N$  bits of the communication [18]. The wiretapper channel of the wiretap channel II is the adversarial analogue of a  $BEC_{p_w}$  with erasure probability  $p_w = (N - t)/N$ . This is exactly the same as the privacy requirement of the binary secret sharing. But in the wiretap channel II model, the main channel is a clear channel. This is corresponding to the special case of binary secret sharing where the reconstruction is only required when all shares are available. The adversarial analogue of the erasure scenario Wyner wiretap channel should have a main channel that erases  $Np_m$  components out of the total  $N$  components, which is the same as the reconstruction

requirement of binary secret sharing. Moreover, same as the Wyner wiretap channel model, the secrecy of the wiretap channel II is only required for uniform message and satisfies the weak (resp., strong) secrecy mentioned above.

## 1.2 Our contributions

We motivate the study of secret sharing scheme with fixed share size  $q$ , and study the extremal case of binary shares. Our goal is to show that even in this extremely restrictive case, a slight relaxation of the privacy and reconstruction notions of ramp secret sharing guarantees explicit construction of families of ramp schemes<sup>2</sup> with any constant relative privacy and reconstruction thresholds  $0 \leq \tau < \rho \leq 1$ , in particular, the relative threshold gap  $\gamma = \rho - \tau$  can be an arbitrarily small constant. Namely, for any constants  $0 \leq \tau < \rho \leq 1$ , it can be guaranteed that any  $\tau N$  or fewer shares reveal essentially no information about the secret, whereas any  $\rho N$  or more shares can reconstruct the exact secret with a negligible failure probability. While we only focus on the extremal special case  $q = 2$  in this presentation, all our results can be extended to any constant  $q$  (see Section 6).

We consider binary sharing of a large  $\ell$ -bit secret and for this work focus on the asymptotic case where the secret length  $\ell$ , and consequently the number of players  $N$ , are sufficiently large. We replace perfect privacy with semantic security, the strongest cryptographic notion of privacy second only to perfect privacy. That is, for any two secrets (possibly chosen by the adversary), we require the adversary's view to be statistically indistinguishable. The view of the adversary is a random variable with randomness coming solely from the internal randomness of the sharing algorithm. The notion of indistinguishability that we use is statistical (total variation) distance bounded by a leakage error parameter  $\varepsilon$  that is negligible in  $N$ . Using non-perfect privacy creates a distinction between non-adaptive and adaptive secrecy. A non-adaptive adversary chooses any  $\tau$  fraction of the  $N$  players at once, and receives their corresponding shares. An adaptive adversary however, selects share holders one by one, receives their shares and uses its available information to make its next choice. When  $\varepsilon = 0$ , i.e., when perfect privacy holds, non-adaptive secrecy automatically implies adaptive secrecy as well. However, this is not necessarily the case when  $\varepsilon > 0$  and we thus study the two cases separately. Similarly, we replace the perfect reconstruction with probabilistic reconstruction allowing a failure probability  $\delta$  that is negligible in  $N$ . The special case of  $\delta = 0$  means perfect reconstruction.

Note that secret sharing with fixed share size necessarily imposes certain restrictions that are not common in standard secret sharing. Unlike secret sharing with share length dependent on the secret length (for threshold schemes) or secret length and threshold gap (for ramp schemes), binary sharing of an  $\ell$ -bit secret obviously requires at least  $\ell$  shares to accommodate the secret information. For a family of ramp secret sharing schemes with fixed share size  $q$  and fixed relative thresholds  $0 \leq \tau < \rho \leq 1$ , as  $N$  grows the absolute gap length  $(\rho - \tau)N$  grows, and the accommodatable length of the secret is expected to grow and so the ratio  $\ell/N \in (0, 1]$  becomes a key parameter of interest for the family, referred to as the *coding rate*. As is customary in coding theory, it is desired to characterize the maximum possible ratio  $\ell/N \in (0, 1]$  for binary secret sharing. We use the relation (a similar relation was used in [10] for robust secret sharing) between a binary secret sharing family with relative threshold  $(\tau, \rho)$  and codes for a Wyner wiretap channel with two BEC's to derive a coding rate upper bound of  $\rho - \tau$  for binary secret sharing (see Lemma 11).

<sup>2</sup> We abuse the notion and will continue calling these relaxed schemes ramp schemes.



Our main technical contributions are explicit constructions of binary secret sharing schemes in both the non-adaptive and adaptive models, and proving optimality of non-adaptive construction. Namely, we prove the following:

► **Theorem 1** (informal summary of Lemma 11, Corollary 17, and Corollary 21). *For any choice of  $0 \leq \tau < \rho \leq 1$ , and large enough  $N$ , there is an explicit construction of a binary secret sharing scheme with  $N$  players that provides (adaptive or non-adaptive) privacy against leakage of any  $\tau N$  or fewer shares, as well as reconstruction from any  $\rho N$  or more of the shares (achieving semantic secrecy with negligible error and imperfect reconstruction with negligible failure probability). For non-adaptive secrecy, the scheme shares a secret of length  $\ell = (\rho - \tau - o(1))N$ , which is asymptotically optimal. For adaptive secrecy, the scheme shares a secret of length  $\ell = \Omega((\rho - \tau)N)$ .*

As a side contribution, our findings unify the Wyner wiretap model and its adversarial analogue. Our capacity-achieving construction of binary secret sharing for non-adaptive adversaries implies that the secrecy capacity of the adversarial analogue of the erasure scenario Wyner wiretap channel is similarly characterized by the erasure ratios of the two channels. Moreover, the secrecy can be strengthened to semantic security.

This answers an open question posted in [1]. The authors studied a generalisation of the wiretap II model, where the adversary chooses  $t$  bits to observe and erases them. They showed that the rate  $1 - \tau - h_2(\tau)$ , where  $h_2(\cdot)$  is the binary entropy function, can be achieved and left open the question of whether a higher rate is achievable. Our result specialized to their setting shows that, the rate  $1 - 2\tau$  can be explicitly achieved.

### 1.3 Our approach and techniques

Our explicit constructions follow the paradigm of invertible randomness extractors formalized in [11]. Invertible extractors were used in [11] for explicit construction of optimal wiretap coding schemes in the Wiretap channel II [18]. This, in particular, is corresponding to the  $\rho = 1$  special case of secret sharing where reconstruction is only required when all shares are available. Moreover, the secrecy there is an information-theoretic notion, and only required to hold for uniform messages. The consequence of the latter is that the construction in [11] does not directly give us binary secret sharing, not even for the  $\rho = 1$  special case. The exposition below is first focused on how semantic security is achieved.

As in [11], we rely on *invertible affine extractors* as our primary technical tool. Such an extractor is an explicit function  $\text{AExt}: \{0, 1\}^n \rightarrow \{0, 1\}^\ell$  such that, for any random variable  $X$  uniformly distributed over an unknown  $k$ -dimensional affine subspace of  $\mathbb{F}_2^n$ , the distribution of  $\text{AExt}(X)$  is close to the uniform distribution over  $\mathbb{F}_2^\ell$  in statistical distance. Furthermore, the invertibility guarantee provides an efficient algorithm for sampling a uniform element from the set  $\text{AExt}^{-1}(s)$  of pre-images for any given output  $s \in \mathbb{F}_2^\ell$ .

It is then natural to consider the affine extractor's uniform inverter as a candidate building block for the sharing algorithm of a secret sharing scheme. Intuitively, if the secret  $s$  is chosen uniformly at random, we have the guarantee that for any choice of a bounded number of the bits of its random pre-image revealed to the adversary, the distribution of the random pre-image conditioned on the revealed value satisfies that of an affine source. Now according to the definition of an affine extractor, the extractor's output (i.e., the secret  $s$ ) remains uniform (and thus unaffected in distribution) given the information revealed to the adversary. Consequently, secrecy should at least hold in an information-theoretic sense, i.e. the mutual information between the secret and the revealed vector components is zero. This is what was formalized and used in [11] for the construction of Wiretap channel II codes.

For non-adaptive adversaries, in fact it is possible to use *invertible seeded extractors* rather than invertible affine extractors described in the above construction. A (strong) seeded extractor assumes, in addition to the main input, an independent seed as an auxiliary input and ensures uniformity of the output for most fixings of the seed. The secret sharing encoder appends a randomly chosen seed to the encoding and inverts the extractor with respect to the chosen seed. Then, the above argument would still hold even if the seed is completely revealed to the adversary.

The interest in the use of seeded, as opposed to seedless affine, extractors is twofold. First, nearly optimal and very efficient constructions of seeded extractors are known in the literature that extract nearly the entire source entropy with only a short seed. This allows us to attain nearly optimal rates for the non-adaptive case. Furthermore, and crucially, such nearly optimal extractor constructions (in particular, Trevisan's extractor [24, 19]) can in fact be linear functions for every fixed choice of the seed (in contrast, seedless affine extractors can never be linear functions). We take advantage of the linearity of the extractor in a crucial way and use a rather delicate analysis to show that in fact the linearity of the extractor can be utilized to prove that the resulting secret sharing scheme provides the stringent worst-case secret guarantee which is a key requirement distinguishing secret sharing schemes (a cryptographic primitive) from wiretap codes (an information-theoretic notion).

Using a seeded extractor instead of a seedless extractor, however, introduces a new challenge. In order for the seeded extractor to work, the seed has to be independent of the main input, which is a distribution induced by the adversary's choice of reading positions. The independence of the seed and the main input can be directly argued when the adversary is non-adaptive. An adaptive adversary, however, may choose its reading positions to learn about the seed first, and then choose the rest of the reading positions according to the value of the seed. In this case, we can not prove the independence of the seed and the main input.

For adaptive adversaries, we go back to using an invertible affine extractor. We prove that both security for worst-case messages and against adaptive adversaries are guaranteed if the affine extractor provides the strong guarantee of having a nearly uniform output with respect to the  $\ell_\infty$  measure rather than  $\ell_1$ . However, this comes at the cost of the extractor not being able to extract the entire entropy of the source, leading to ramp secret sharing schemes with slightly sub-optimal rates, albeit still achieving rates within a constant factor of the optimum. As a proof of concept, we utilize a simple padding and truncation technique to convert any off-the-shelf seedless affine extractor (such as those of Bourgain [7] or Li [16]) to one that satisfies the stronger uniformity condition that we require.

We now turn to reconstruction from an incomplete set of shares. In order to provide reconstructibility from a subset of size  $r$  of the shares, we naturally compose the encoding obtained from the extractor's inversion routine with a linear erasure-correcting code. The linearity of the code ensures that the extractor's input subject to the adversary's observation (which now can consist of linear combinations of the original encoding) remains uniform on some affine space, thus preserving the privacy guarantee.

However, since by the known rate-distance trade-offs of binary error-correcting codes, no deterministic coding scheme can correct more than a  $1/2$  fraction of erasures (a constraint that would limit the choice of  $\rho$ ), the relaxed notion of *stochastic coding schemes* is necessary for us to allow reconstruction for all choices of  $\rho \in (\tau, 1]$ . Intuitively, a stochastic code is a randomized encoder with a deterministic decoder, that allows the required fraction of errors to be corrected. We utilize what we call a *stochastic affine* code. Such codes are equipped with encoders that are affine functions of the message for every fixing of the encoder's internal randomness. We show that such codes are as suitable as deterministic linear codes for providing the linearity properties that our construction needs.

In fact, we need capacity-achieving stochastic erasure codes, i.e., those that correct every  $1 - \rho$  fraction of erasures at asymptotic rate  $\rho$ , to be able to construct binary secret sharing schemes with arbitrarily small relative gap  $\gamma = \rho - \tau$ . To construct capacity-achieving stochastic affine erasure codes, we utilize a construction of stochastic codes due to Guruswami and Smith [15] for bit-flip errors. We observe that this construction can be modified to yield capacity-achieving erasure codes. Roughly speaking, this is achieved by taking an explicit capacity-achieving linear code for BEC and pseudo-randomly shuffling the codeword positions. Combined with a delicate encoding of hidden “control information” to communicate the choice of the permutation to the decoder in a robust manner, the construction transforms robustness against random erasures to worst-case erasures at the cost of making the encoder randomized.

## 1.4 Organization of the paper

Section 2 contains a brief introduction to the two building blocks for our constructions: randomness extractors and stochastic codes. In Section 3, we formally define the binary secret sharing model and prove a coding rate upper bound. Section 4 contains a capacity-achieving construction with privacy against non-adaptive adversaries. Section 5 contains a constant rate construction with privacy against adaptive adversaries. Finally, we conclude the paper and discuss open problems in Section 6.

## 2 Preliminaries and definitions

In this section, we review the necessary facts and results about randomness extractors, both the seeded and seedless affine variants, as well as the stochastic erasure correcting codes.

Randomness extractors extract close to uniform bits from input sequences that are not uniform but have some guaranteed entropy. The closeness to uniform of the extractor output is measured by the statistical distance (half the  $\ell_1$ -norm). For a set  $\mathcal{X}$ , we use  $X \leftarrow \mathcal{X}$  to denote that  $X$  is distributed over the set  $\mathcal{X}$ . For two random variables  $X, Y \leftarrow \mathcal{X}$ , the statistical distance between  $X$  and  $Y$  is defined as,

$$\text{SD}(X; Y) = \frac{1}{2} \sum_{x \in \mathcal{X}} |\Pr[X = x] - \Pr[Y = x]|.$$

We say  $X$  and  $Y$  are  $\varepsilon$ -close if  $\text{SD}(X, Y) \leq \varepsilon$ . A *randomness source* is a random variable with lower bound on its min-entropy, which is defined by  $H_\infty(X) = -\log \max_x \{\Pr[X = x]\}$ . We say a random variable  $X \leftarrow \{0, 1\}^n$  is a  $(n, k)$ -source if  $H_\infty(X) \geq k$ .

For well structured sources, there exist deterministic functions that can extract close to uniform bits. The *support* of  $X \leftarrow \mathcal{X}$  is the set of  $x \in \mathcal{X}$  such that  $\Pr[X = x] > 0$ . An *affine*  $(n, k)$ -source is an  $(n, k)$ -source whose support is an affine sub-space of  $\{0, 1\}^n$  and each vector in the support occurs with the same probability. Let  $U_m$  denote the random variable uniformly distributed over  $\{0, 1\}^m$ .

► **Definition 2.** A function  $\text{AExt}: \{0, 1\}^n \rightarrow \{0, 1\}^m$  is an affine  $(k, \varepsilon)$ -extractor if for any affine  $(n, k)$ -source  $X$ , we have

$$\text{SD}(\text{AExt}(X); U_m) \leq \varepsilon.$$

An affine extractor can not be a linear function.

For general  $(n, k)$ -sources, there does not exist a deterministic function that can extract close to uniform bits from all of them simultaneously. A family of deterministic functions are needed.

► **Definition 3.** A function  $\text{Ext}: \{0, 1\}^d \times \{0, 1\}^n \rightarrow \{0, 1\}^m$  is a strong seeded  $(k, \varepsilon)$ -extractor if for any  $(n, k)$ -source  $X$ , we have

$$\text{SD}(S, \text{Ext}(S, X); S, U_m) \leq \varepsilon,$$

where  $S$  is chosen uniformly from  $\{0, 1\}^d$ . A seeded extractor  $\text{Ext}(\cdot, \cdot)$  is called linear if for any fixed seed  $S = s$ , the function  $\text{Ext}(s, \cdot)$  is a linear function.

We will use Trevisan's extractor [24] in our first construction. In particular, we use the following improvement of this extractor due to Raz, Reingold and Vadhan [19].

► **Lemma 4** ([19]). *There is an explicit linear strong seeded  $(k, \varepsilon)$ -extractor  $\text{Ext}: \{0, 1\}^d \times \{0, 1\}^n \rightarrow \{0, 1\}^\ell$  with  $d = O(\log^3(n/\varepsilon))$  and  $\ell = k - O(d)$ .*

We will use Bourgain's affine extractor in our second construction. We note, however, that we could have used other explicit extractors for this purpose, such as [16].

► **Lemma 5** ([7]). *For every constant  $0 < \mu \leq 1$ , there is an explicit affine  $(\mu n, \varepsilon)$ -extractor  $\text{AExt}: \{0, 1\}^n \rightarrow \{0, 1\}^m$  with output length  $m = \Omega(n)$  and error  $\varepsilon = 2^{-\Omega(n)}$ .*

Explicit constructions of randomness extractors have efficient forward direction of extraction. In some applications, we usually need to efficiently invert the process: Given an extractor output, sample a random pre-image.

► **Definition 6** ([11]). Let  $f$  be a mapping from  $\{0, 1\}^n$  to  $\{0, 1\}^m$ . For  $0 \leq v < 1$ , a function  $\text{Inv}: \{0, 1\}^m \times \{0, 1\}^r \rightarrow \{0, 1\}^n$  is called a  $v$ -inverter for  $f$  if the following conditions hold:

- (Inversion) Given  $y \in \{0, 1\}^m$  such that its pre-image  $f^{-1}(y)$  is nonempty, for every  $r \in \{0, 1\}^r$  we have  $f(\text{Inv}(y, r)) = y$ .
- (Uniformity)  $\text{Inv}(U_m, U_r)$  is  $v$ -close to  $U_n$ .

A  $v$ -inverter is called efficient if there is a randomized algorithm that runs in worst-case polynomial time and, given  $y \in \{0, 1\}^m$  and  $r$  as a random seed, computes  $\text{Inv}(y, r)$ . We call a mapping  $v$ -invertible if it has an efficient  $v$ -inverter, and drop the prefix  $v$  from the notation when it is zero. We abuse the notation and denote the inverter of  $f$  by  $f^{-1}$ .

A *stochastic code* has a randomised encoder and a deterministic decoder. The encoder  $\text{Enc}: \{0, 1\}^m \times \mathcal{R} \rightarrow \{0, 1\}^n$  uses local randomness  $R \leftarrow \mathcal{R}$  to encode a message  $m \in \{0, 1\}^m$ . The decoder is a deterministic function  $\text{Dec}: \{0, 1\}^n \rightarrow \{0, 1\}^m \cup \{\perp\}$ . The decoding probability is defined over the encoding randomness  $R \leftarrow \mathcal{R}$ . Stochastic codes are known to explicitly achieve the capacity of some adversarial channels [15].

Affine sources play an important role in our constructions. We define a general requirement for the stochastic code used in our constructions.

► **Definition 7** (Stochastic Affine codes). Let  $\text{Enc}: \{0, 1\}^m \times \mathcal{R} \rightarrow \{0, 1\}^n$  be the encoder of a stochastic code. We say it is a stochastic affine code if for any  $r \in \mathcal{R}$ , the encoding function  $\text{Enc}(\cdot, r)$  specified by  $r$  is an affine function of the message. That is we have

$$\text{Enc}(m, r) = mG_r + \Delta_r,$$

where  $G_r \in \{0, 1\}^{m \times n}$  and  $\Delta_r \in \{0, 1\}^n$  are specified by the randomness  $r$ .

We then adapt a construction in [15] to obtain the following capacity-achieving Stochastic Affine-Erasure Correcting Code (SA-ECC). In particular, we show for any  $p \in [0, 1)$ , there is an explicit stochastic affine code that corrects  $p$  fraction of adversarial erasures and achieves the rate  $1 - p$ .

► **Lemma 8** (Adapted from [15]). *For every  $p \in [0, 1)$ , and every  $\xi > 0$ , there is an efficiently encodable and decodable stochastic affine code  $(\text{Enc}, \text{Dec})$  with rate  $R = 1 - p - \xi$  such that for every  $\mathbf{m} \in \{0, 1\}^{NR}$  and erasure pattern of at most  $p$  fraction, we have  $\Pr[\text{Dec}(\widetilde{\text{Enc}}(\mathbf{m})) = \mathbf{m}] \geq 1 - \exp(-\Omega(\xi^2 N / \log^2 N))$ , where  $\widetilde{\text{Enc}}(\mathbf{m})$  denotes the partially erased random codeword and  $N$  denotes the length of the codeword.*

**Proof Sketch.** We defer the details of the construction to the full version of this paper. For now we refer to [15, Theorem 6.1] and point out the adaptations needed. There are six building blocks involved in the construction: SC, RS, Samp, KNR, POLY<sub>*t*</sub> and REC. We replace the first and last building blocks.

The first building block is a *Stochastic Code* (SC). We need two properties from this building block: detect when the codeword is masked by a random offset and correct from erasures of no more than  $1 - \rho + \xi$  fraction. Since the correction is with respect to erasure, as opposed to flips in [15, Theorem 6.1], we then use an erasure list-decodable code [14] instead of an error list-decodable code. This is necessary when  $\rho < \frac{1}{2}$ , since the latter can not correct errors beyond one half.

The last building block is a *Random Error Code* (REC). We need two properties from this building block: correct from random erasures of  $1 - \rho$  fraction and the encoder is a linear function. The linearity was not required in [15, Theorem 6.1]. Explicit linear codes at rate  $1 - p$  that correct  $p$  fraction of random erasures are known. We can use any explicit construction of capacity achieving linear codes for BEC <sub>$1-\rho$</sub>  to obtain REC and use a similar argument of [22]. ◀

### 3 Binary secret sharing schemes

In this section, we define our model of nearly-threshold binary secret sharing schemes. We begin with a description of the two models of non-adaptive and adaptive adversaries which can access up to  $t$  of the  $N$  shares.

A *leakage oracle* is a machine  $\mathcal{O}(\cdot)$  that takes as input an  $N$ -bit string  $\mathbf{c} \in \{0, 1\}^N$  and then answers the *leakage queries* of the type  $I_j$ , for  $I_j \subset [N]$ ,  $j = 1, 2, \dots, q$ . Each query  $I_j$  is answered with  $\mathbf{c}_{I_j}$ . An interactive machine  $\mathcal{A}$  that issues the leakage queries is called a *leakage adversary*. Let  $A_{\mathbf{c}} = \cup_{j=1}^q I_j$  denote the union of all the index sets chosen by  $\mathcal{A}$  when the oracle input is  $\mathbf{c}$ . The oracle is called  $t$ -bounded, denoted by  $\mathcal{O}_t(\cdot)$ , if it rejects leakage queries from  $\mathcal{A}$  if there exists some  $\mathbf{c} \in \{0, 1\}^N$  such that  $|A_{\mathbf{c}}| > t$ . An adaptive leakage adversary decides the index set  $I_{j+1}$  according to the oracle's answers to all previous queries  $I_1, \dots, I_j$ . A non-adaptive leakage adversary has to decide the index set  $A_{\mathbf{c}}$  before any information about  $\mathbf{c}$  is given. This means that for a non-adaptive adversary, given any oracle input  $\mathbf{c} \in \{0, 1\}^N$ , we always have  $A_{\mathbf{c}} = A$  for some  $A \subset [N]$ . Let  $\text{View}_{\mathcal{A}}^{\mathcal{O}_t(\cdot)}$  denote the view of the leakage adversary  $\mathcal{A}$  interacting with a  $t$ -bounded leakage oracle. When  $\mathcal{A}$  is non-adaptive, we use the shorthand  $\text{View}_{\mathcal{A}}^{\mathcal{O}_t(\cdot)} = (\cdot)_A$ , for some  $A \subset [N]$  of size  $|A| \leq t$ .

A function  $\varepsilon: \mathbb{N} \rightarrow \mathbb{R}$  is called *negligible* if for every positive integer  $k$ , there exists an  $N_k \in \mathbb{N}$  such that  $|\varepsilon(N)| < \frac{1}{N^k}$  for all  $N > N_k$ . The following definition of ramp Secret Sharing Scheme (SSS) allows imperfect privacy and reconstruction with errors bounded by negligible functions  $\varepsilon(\cdot)$  and  $\delta(\cdot)$ , respectively.

► **Definition 9.** For any  $0 \leq \tau < \rho \leq 1$ , an  $(\varepsilon(N), \delta(N))$ -SSS with relative threshold pair  $(\tau, \rho)$  is a pair of polynomial-time algorithms (Share, Recst),

$$\text{Share}: \{0, 1\}^{\ell(N)} \times \mathcal{R} \rightarrow \{0, 1\}^N,$$

where  $\mathcal{R}$  denote the randomness set, and

$$\text{Recst}: \widetilde{\{0, 1\}^N} \rightarrow \{0, 1\}^{\ell(N)} \cup \{\perp\},$$

where  $\widetilde{\{0, 1\}}^N$  denotes the subset of  $(\{0, 1\} \cup \{?\})^N$  with at least  $N\rho$  components not equal to the erasure symbol “?”, that satisfy the following properties.

- Reconstruction: Given  $r(N) = N\rho$  correct shares of a share vector  $\text{Share}(\mathbf{s})$ , the reconstruction algorithm  $\text{Recst}$  reconstructs the secret  $\mathbf{s}$  with probability at least  $1 - \delta(N)$ .

When  $\delta(N) = 0$ , we say the SSS has perfect reconstruction.

- Privacy (non-adaptive/adaptive):

- Non-adaptive: for any  $\mathbf{s}_0, \mathbf{s}_1 \in \{0, 1\}^{\ell(N)}$ , any  $A \subset [N]$  of size  $|A| \leq t(N) = N\tau$ ,

$$\text{SD}(\text{Share}(\mathbf{s}_0)_A; \text{Share}(\mathbf{s}_1)_A) \leq \varepsilon(N). \quad (3)$$

- Adaptive: for any  $\mathbf{s}_0, \mathbf{s}_1 \in \{0, 1\}^{\ell(N)}$  and any adaptive adversary  $\mathcal{A}$  interacting with a  $t(N)$ -bounded leakage oracle  $\mathcal{O}_{t(N)}(\cdot)$  for  $t(N) = N\tau$ ,

$$\text{SD}\left(\text{View}_{\mathcal{A}}^{\mathcal{O}_{t(N)}(\text{Share}(\mathbf{s}_0))}; \text{View}_{\mathcal{A}}^{\mathcal{O}_{t(N)}(\text{Share}(\mathbf{s}_1))}\right) \leq \varepsilon(N). \quad (4)$$

When  $\varepsilon(N) = 0$ , we say the SSS has perfect privacy.

The difference  $\gamma = \rho - \tau$  is called the relative gap, since  $N\gamma = r(N) - t(N)$  is the threshold gap of the scheme. When clear from context, we write  $\varepsilon, \delta, t, k, \ell$  instead of  $\varepsilon(N), \delta(N), t(N), r(N), \ell(N)$ . When the parameters are not specified, we call a  $(\varepsilon, \delta)$ -SSS simply a binary SSS.

In the above definition, a binary SSS has a pair of designed relative thresholds  $(\tau, \rho)$ . In this work, we are concerned with constructing nearly-threshold binary SSS, namely, binary SSS with arbitrarily small relative gap  $\gamma = \rho - \tau$ . We also want our binary SSS to share a large secret  $\ell = \Omega(N)$ .

► **Definition 10.** For any  $0 \leq \tau < \rho \leq 1$ , a coding rate  $R \in [0, 1]$  is achievable if there exists a family of  $(\varepsilon, \delta)$ -SSS with relative threshold pair  $(\tau, \rho)$  such that  $\varepsilon$  and  $\delta$  are both negligible in  $N$  and  $\frac{\ell}{N} \rightarrow R$ . The highest achievable coding rate of binary SSS for a pair  $(\tau, \rho)$  is called its capacity.

By relating binary SSS to Wyner wiretap codes with a pair of BEC’s, we obtain the following coding rate upper bound for binary SSS.

► **Lemma 11.** For  $0 \leq \tau < \rho \leq 1$ , the coding rate capacity of binary SSS with relative threshold pair  $(\tau, \rho)$  is asymptotically upper-bounded by  $\rho - \tau$ .

Due to the space constraint, details of the proof are given in the full version of this paper.

**Proof Sketch.** Let  $(\text{Share}, \text{Recst})$  be a non-adaptive binary SSS with relative threshold pair  $(\tau, \rho)$ . We use  $\text{Share}$  as the encoder and  $\text{Recst}$  as the decoder, and verify that we obtain a Wyner wiretap code with a  $\text{BEC}_{p_m}$  main channel and a  $\text{BEC}_{p_w}$  wiretapper channel, where  $p_m = 1 - \rho - \xi$  and  $p_w = 1 - \tau + \xi$ , respectively, for arbitrarily small  $\xi > 0$ . Erasures in binary SSS are worst-case, while they are probabilistic in the Wyner wiretap model. We however note that asymptotically, the number of random erasures of  $\text{BEC}_{p_m}$  and  $\text{BEC}_{p_w}$  approaches  $Np_m$  and  $Np_w$ , respectively, with overwhelming probability, and so a code that protects against worst-case erasure can be used as a code with probabilistic erasure. We also take into account the difference in the secrecy notion in binary SSS and Wyner wiretap code. ◀

In the rest of the paper, we give two constant rate constructions of nearly-threshold binary SSS against non-adaptive adversary and adaptive adversary, respectively. The non-adaptive adversary construction is optimal in the sense that the coding rate achieves the upper bound in Lemma 11.



#### 4 Secret sharing against non-adaptive adversaries

We first present our construction of capacity-achieving binary SSS against non-adaptive adversaries, using linear strong seeded extractors and optimal rate stochastic erasure correcting codes. The following theorem describes the construction using these components.

► **Theorem 12.** *Let  $\text{Ext}: \{0, 1\}^d \times \{0, 1\}^n \rightarrow \{0, 1\}^\ell$  be a linear strong seeded  $(n - t, \frac{\varepsilon}{8})$ -extractor and  $\text{Ext}^{-1}(\mathbf{z}, \cdot): \{0, 1\}^\ell \times \mathcal{R}_1 \rightarrow \{0, 1\}^n$  be the inverter of the function  $\text{Ext}(\mathbf{z}, \cdot)$  that maps an  $\mathbf{s} \in \{0, 1\}^\ell$  to one of its pre-images chosen uniformly at random. Let  $(\text{SA-ECCenc}, \text{SA-ECCdec})$  be a stochastic affine-erasure correcting code with the encoder  $\text{SA-ECCenc}: \{0, 1\}^{d+n} \times \mathcal{R}_2 \rightarrow \{0, 1\}^N$  that tolerates  $N - r$  erasures and decodes with success probability at least  $1 - \delta$ . Then the following coding scheme  $(\text{Share}, \text{Recst})$  is a non-adaptive  $(\varepsilon, \delta)$ -SSS with threshold pair  $(t, r)$ .*

$$\begin{cases} \text{Share}(\mathbf{s}) &= \text{SA-ECCenc}(\mathbf{Z} \parallel \text{Ext}^{-1}(\mathbf{Z}, \mathbf{s})), \text{ where } \mathbf{Z} \stackrel{\$}{\leftarrow} \{0, 1\}^d; \\ \text{Recst}(\tilde{\mathbf{v}}) &= \text{Ext}(\mathbf{z}, \mathbf{x}), \text{ where } (\mathbf{z} \parallel \mathbf{x}) = \text{SA-ECCdec}(\tilde{\mathbf{v}}). \end{cases}$$

Here  $\tilde{\mathbf{v}}$  denotes an incomplete version of a share vector  $\mathbf{v} \in \{0, 1\}^N$  with some of its components replaced by erasure symbols.

The proof of Theorem 12 will follow naturally from Lemma 13. We first state and prove this general property of a linear strong extractor, which is of independent interest. For the property to hold, we in fact only need the extractor to be able to extract from affine sources. The proof of Lemma 13 is a bit long. We then break it into a claim and two propositions.

► **Lemma 13.** *Let  $\text{Ext}: \{0, 1\}^d \times \{0, 1\}^n \rightarrow \{0, 1\}^m$  be a linear strong  $(k, \varepsilon)$ -extractor. Let  $f_A: \{0, 1\}^{d+n} \rightarrow \{0, 1\}^t$  be any affine function with output length  $t \leq n - k$ . For any  $\mathbf{m}, \mathbf{m}' \in \{0, 1\}^m$ , let  $(\mathbf{Z}, \mathbf{X}) = (\mathbf{U}_d, \mathbf{U}_n) \mid (\text{Ext}(\mathbf{U}_d, \mathbf{U}_n) = \mathbf{m})$  and  $(\mathbf{Z}', \mathbf{X}') = (\mathbf{U}_d, \mathbf{U}_n) \mid (\text{Ext}(\mathbf{U}_d, \mathbf{U}_n) = \mathbf{m}')$ . We have*

$$\text{SD}(f_A(\mathbf{Z}, \mathbf{X}); f_A(\mathbf{Z}', \mathbf{X}')) \leq 8\varepsilon. \quad (5)$$

**Proof.** For the above pairwise guarantee (5) to hold, it suffices to show that for every fixed choice of  $\mathbf{m} \in \{0, 1\}^m$ , the distribution of  $f_A(\mathbf{Z}, \mathbf{X})$  is  $(4\varepsilon)$ -close to  $\mathbf{U}_d \times \mathcal{D}$ , where  $\mathbf{U}_d$  is the uniform distribution on  $\{0, 1\}^d$ .

Without loss of generality, we assume that the linear function  $\text{Ext}(\mathbf{z}, \cdot): \{0, 1\}^n \rightarrow \{0, 1\}^m$ , for every seed  $\mathbf{z}$ , has the entire  $\{0, 1\}^m$  as its image<sup>3</sup>. Without loss of generality, it suffices to assume that  $f_A$  is of the form  $f_A(\mathbf{Z}, \mathbf{X}) = (\mathbf{Z}, W(\mathbf{X}))$  for some affine function  $W: \{0, 1\}^n \rightarrow \{0, 1\}^t$ . This is because for any arbitrary  $f_A$ , the information contained in  $f_A(\mathbf{Z}, \mathbf{X})$  can be obtained from  $(\mathbf{Z}, W(\mathbf{X}))$  for a suitable choice of  $W$ . Let  $\mathcal{D}$  be the uniform distribution on the image of  $W$ .

Let  $\mathbf{K} \leftarrow \{0, 1\}^n$  be a random variable uniformly distributed over the kernel of the linear transformation defined by  $W$ , and note that it has entropy at least  $n - t \geq k$ . The extractor  $\text{Ext}$  thus guarantees that  $\text{Ext}(\mathbf{Z}, \mathbf{K})$ , for a uniform and independent seed  $\mathbf{Z}$ , is  $\varepsilon$ -close to uniform. By averaging, it follows that for at least  $1 - 4\varepsilon$  fraction of the choices of the seed  $\mathbf{z} \in \{0, 1\}^d$ , the distribution of  $\text{Ext}(\mathbf{z}, \mathbf{K})$  is  $(1/4)$ -close to uniform.

<sup>3</sup> If this condition is not satisfied for some choice  $\mathbf{z}$  of the seed, there must be linear dependencies between the  $m$  output bits of  $\text{Ext}(\mathbf{z}, \cdot)$ . Therefore, for this choice  $\text{Ext}(\mathbf{z}, \cdot)$  can never be an extractor and arbitrarily changing  $\text{Ext}(\cdot, \mathbf{z})$  to be an arbitrary full rank linear function will not change the overall performance of the extractor.



► **Claim 14.** *Let  $U$  be uniformly distributed on  $\{0, 1\}^m$  and  $U'$  be any affine source that is not uniform on  $\{0, 1\}^m$ . Then, the statistical distance between  $U$  and  $U'$  is at least  $1/2$ .*

Claim 14 follows from the observation that any affine source  $U'$  that is not uniform on  $\{0, 1\}^m$  will have a support (the set of vectors  $u$  such that  $\Pr[U' = u] > 0$ ) that is an affine subspace of  $\{0, 1\}^m$  with dimension at most  $m - 1$ .

Continuing with the previous argument, since  $\text{Ext}$  is a linear function for every seed, the distribution of  $\text{Ext}(z, K)$  for any seed  $z$  is an affine source. Therefore, the above claim allows us to conclude that for at least  $1 - 4\varepsilon$  fraction of the choices of  $z$ , the distribution of  $\text{Ext}(z, K)$  is *exactly* uniform. Let  $\mathcal{G} \subseteq \{0, 1\}^d$  be the set of such choices of the seed. Observe that if  $\text{Ext}(z, K)$  is uniform for some seed  $z$ , then for any affine translation of  $K$ , namely,  $K + v$  for any  $v \in \{0, 1\}^n$ , we have that  $\text{Ext}(z, K + v)$  is uniform as well. This is due to the linearity of the extractor.

Recall that our goal is to show that  $f_A(Z, X) = (Z, W(X))$  is  $(4\varepsilon)$ -close to  $U_d \times \mathcal{D}$ . The distribution  $(Z, W(X))$  is obtained as  $(U_d, W(U_n)) | (\text{Ext}(U_d, U_n) = m)$ . For the rest of the proof, we first find out the distribution  $(U_d, W(U_n)) | (\text{Ext}(U_d, U_n) = m, U_d = z)$  for a seed  $z \in \mathcal{G}$  (Proposition 15) and then take the convex combination over the uniform seed to obtain  $(Z, W(X))$  (Proposition 16). The argument starts with a uniform message  $M \stackrel{\$}{\leftarrow} \{0, 1\}^m$  instead of a particular message  $m \in \{0, 1\}^m$ . We define a new set of simplified notations. Let  $Z \stackrel{\$}{\leftarrow} \{0, 1\}^d$  be an independent and uniform seed. Let  $(Z, Y)$  be the pre-image of  $M$  and  $(Z, W) := f_A(Z, Y)$ . Proposition 15 and 16 are stated in terms of the triple  $(M, Z, W)$ .

► **Proposition 15.** *Let  $z \in \mathcal{G}$  and consider any  $m \in \{0, 1\}^m$ . Then, the conditional distribution of  $W | (Z = z, M = m)$  is exactly  $\mathcal{D}$ .*

To prove Proposition 15, note that the distribution of  $(Z, Y)$  is uniform on  $\{0, 1\}^{d+n}$ . Now, fix any  $z \in \mathcal{G}$  and let  $w \in \{0, 1\}^t$  be any element in the image of  $W(\cdot)$ . Since the conditional distribution  $Y | (Z = z)$  is uniform over  $\{0, 1\}^n$ , further conditioning on  $W(Y) = w$  yields that  $Y | (Z = z, W = w)$  is uniform over a translation of the kernel of  $W(\cdot)$ . By the assumption  $z \in \mathcal{G}$  and recalling  $M = \text{Ext}(Z, Y)$ , we therefore know that the extractor output is exactly uniform over  $\{0, 1\}^m$ . That is,  $M | (Z = z, W = w)$  is exactly uniform over  $\{0, 1\}^m$  and hence in this case  $M$  and  $W$  are independent. On the other hand, the distribution of  $(Z, W)$  is exactly  $U_d \times \mathcal{D}$ , since the map  $W(\cdot)$  is linear. In particular, for any  $z \in \{0, 1\}^d$ , the conditional distribution  $W | (Z = z)$  is exactly  $\mathcal{D}$ . This together with the fact that  $M$  and  $W$  are independent yield that the conditional distribution of  $(M, W) | (Z = z)$  is exactly  $U_m \times \mathcal{D}$ . We have therefore proved Proposition 15.

► **Proposition 16.** *For any  $m \in \{0, 1\}^m$ , the conditional distribution of  $(Z, W) | (M = m)$  is  $(4\varepsilon)$ -close to  $U_d \times \mathcal{D}$ .*

To prove Proposition 16, it suffices to note that the distribution of  $(Z, W) | (M = m)$  is a convex combination of the distributions  $(Z, W) | (M = m, Z = z)$  and then use the result of Proposition 15 along with the fact that  $\Pr[Z \notin \mathcal{G}] \leq 4\varepsilon$ . A detailed derivation follows.

Recall that for any  $z \in \{0, 1\}^d$ , the conditional distribution of  $W | (Z = z)$  is exactly  $\mathcal{D}$  (since  $Y | (Z = z)$  is uniform over  $\{0, 1\}^n$ ). Consider any event  $\mathcal{E} \subseteq \{0, 1\}^{d+t}$  and let  $p := \Pr[(Z, W) \in \mathcal{E}]$ . Since  $Z$  and  $W$  are independent, we have that

$$p = 2^{-d} \sum_{(z, w) \in \mathcal{E}} \mathcal{D}(w),$$

where  $\mathcal{D}(w)$  denotes the probability assigned to the outcome  $w$  by  $\mathcal{D}$ . On the other hand, we shall write down the same probability in the conditional probability space  $M = m$  and show

53:14 Secret Sharing with Binary Shares

that it is different from  $p$  by at most  $4\varepsilon$ , concluding the claim on the statistical distance. We have

$$\begin{aligned} \Pr[(Z, W) \in \mathcal{E} | M = m] &= \sum_{(z,w) \in \mathcal{E}} \Pr[Z = z, W = w | M = m] \\ &= \sum_{(z,w) \in \mathcal{E}, z \in \mathcal{G}} \Pr[Z = z, W = w | M = m] \\ &\quad + \sum_{(z,w) \in \mathcal{E}, z \notin \mathcal{G}} \Pr[Z = z, W = w | M = m]. \end{aligned}$$

Note that

$$\eta := \sum_{(z,w) \in \mathcal{E}, z \notin \mathcal{G}} \Pr[Z = z, W = w | M = m] \leq \Pr[Z \notin \mathcal{G} | M = m] \leq 4\varepsilon,$$

since  $M$  and  $Z$  are independent. Therefore,

$$\begin{aligned} \Pr[(Z, W) \in \mathcal{E} | M = m] &= \sum_{(z,w) \in \mathcal{E}, z \in \mathcal{G}} \Pr[Z = z, W = w | M = m] + \eta \\ &= 2^{-d} \sum_{(z,w) \in \mathcal{E}, z \in \mathcal{G}} \Pr[W = w | M = m, Z = z] + \eta \end{aligned} \tag{6}$$

$$= 2^{-d} \sum_{(z,w) \in \mathcal{E}, z \in \mathcal{G}} \mathcal{D}(w) + \eta \tag{7}$$

$$= 2^{-d} \left( \sum_{(z,w) \in \mathcal{E}} \mathcal{D}(w) - \sum_{(z,w) \in \mathcal{E}, z \notin \mathcal{G}} \mathcal{D}(w) \right) + \eta$$

where (6) uses the independence of  $W$  and  $Z$  and (7) follows from Proposition 15. Observe that

$$\eta' := 2^{-d} \sum_{(z,w) \in \mathcal{E}, z \notin \mathcal{G}} \mathcal{D}(w) = 2^{-d} \sum_{z \notin \mathcal{G}} \sum_{\substack{w: \\ (z,w) \in \mathcal{E}}} \mathcal{D}(w) \leq 2^{-d} (2^d - |\mathcal{G}|) \leq 4\varepsilon.$$

Therefore,

$$\Pr[(Z, W) \in \mathcal{E} | M = m] = p + \eta - \eta' = p \pm 4\varepsilon = \Pr[(Z, W) \in \mathcal{E}] \pm 4\varepsilon,$$

since  $0 \leq \eta \leq 4\varepsilon$  and  $0 \leq \eta' \leq 4\varepsilon$ . We have therefore proved Proposition 16.  $\blacktriangleleft$

With Lemma 13 at hand, we are now at a good position to prove Theorem 12.

**Proof of Theorem 12.** The reconstruction from  $r$  shares follows trivially from the definition of stochastic erasure correcting code. We now prove the privacy.

The sharing algorithm of the SSS (before applying the stochastic affine code) takes a secret, which is a particular extractor output  $\mathbf{s} \in \{0, 1\}^\ell$ , and uniformly samples a seed  $\mathbf{z} \in \{0, 1\}^d$  of  $\text{Ext}$  before uniformly finds an  $\mathbf{x} \in \{0, 1\}^n$  such that  $\text{Ext}(\mathbf{z}, \mathbf{x}) = \mathbf{s}$ . This process of obtaining  $(\mathbf{z}, \mathbf{x})$  is the same as sampling  $(U_d, U_n) \stackrel{\$}{\leftarrow} \{0, 1\}^{d+n}$  and then restrict to  $\text{Ext}(U_d, U_n) = \mathbf{s}$ . We define the random variable tuple

$$(Z, X) := (U_d, U_n) | (\text{Ext}(U_d, U_n) = \mathbf{s}) \tag{8}$$

and refer to it as the pre-image of  $\mathbf{s}$ .

Let  $\Pi_A : \{0, 1\}^N \rightarrow \{0, 1\}^t$  be the projection function that maps a share vector to the  $t$  shares with index set  $A \subset [N]$  chosen by the non-adaptive adversary. Observe that the combination  $(\Pi_A \circ \text{SA-ECCenc}) : \{0, 1\}^{d+n} \rightarrow \{0, 1\}^t$  (for any fixed randomness  $r$  of SA-ECCenc) is an affine function. So the view of the adversary is simply the output of the affine function  $f_A = (\Pi_A \circ \text{SA-ECCenc})$  applied to the random variable tuple  $(Z, X)$  defined in (8).

We can now formulate the privacy of the SSS in this context. We want to prove that the statistical distance of the views of the adversary for a pair of secrets  $s$  and  $s'$  can be made arbitrarily small. The views of the adversary are the outputs of the affine function  $f_A$  with inputs  $(Z, X)$  and  $(Z', X')$  for the secret  $s$  and  $s'$ , respectively. According to Lemma 13, we then have that the privacy error is  $8 \times \frac{\varepsilon}{8} = \varepsilon$ .  $\blacktriangleleft$

We now analyze the coding rate of the  $(\varepsilon, \delta)$ -SSS with relative threshold pair  $(\frac{t}{N}, \frac{r}{N})$  constructed in Theorem 12 when instantiated with the SA-ECC from Lemma 8 and the Ext from Lemma 4. The secret length is  $\ell = n - t - O(d)$ , where the seed length is  $d = O(\log^3(2n/\varepsilon))$ . The SA-ECC encodes  $d + n$  bits to  $N$  bits and with coding rate  $R_{ECC} = \rho - \xi$  for a small  $\xi$  determined by  $\delta$  (satisfying the relation  $\delta = \exp(-\Omega(\xi^2 N / \log^2 N))$  according to Lemma 8). We then have  $n = N(\rho - \xi) - d$ , resulting in the coding rate

$$R = \frac{\ell}{N} = \frac{n - t - O(d)}{N} = \frac{N(\rho - \xi) - t - O(d)}{N} = \rho - \tau - \left(\xi + \frac{O(d)}{N}\right) = \rho - \tau - o(1).$$

► **Corollary 17.** *For any  $0 \leq \tau < \rho \leq 1$ , there is an explicit construction of non-adaptive  $(\varepsilon, \delta)$ -SSS with relative threshold pair  $(\tau, \rho)$  achieving coding rate  $\rho - \tau - o(1)$ , where  $\varepsilon$  and  $\delta$  are both negligible.*

The binary SSS obtained in Corollary 17 is asymptotically optimal as it achieves the upper bound in Lemma 11.

## 5 Secret sharing against adaptive adversaries

In this section, we will describe our construction which achieves privacy against adaptive adversaries, using seedless affine extractors. We start with the specific extraction property needed from our affine extractors.

► **Definition 18.** An affine extractor  $\text{AExt} : \{0, 1\}^n \rightarrow \{0, 1\}^m$  is called  $(k, \varepsilon)$ -almost perfect if for any affine  $(n, k)$ -source  $X$ ,

$$\left| \Pr[\text{AExt}(X) = y] - \frac{1}{2^m} \right| \leq 2^{-m} \cdot \varepsilon, \text{ for any } y \in \{0, 1\}^m.$$

Almost perfect property can be trivially achieved by requiring an exponentially (in  $m$ ) small error in statistical distance, using the relation between  $\ell_\infty$ -norm and  $\ell_1$ -norm.

► **Theorem 19.** *Let  $\text{AExt} : \{0, 1\}^n \rightarrow \{0, 1\}^\ell$  be an invertible  $(n - t, \frac{\varepsilon}{2})$ -almost perfect affine extractor and  $\text{AExt}^{-1} : \{0, 1\}^\ell \times \mathcal{R}_1 \rightarrow \{0, 1\}^n$  be its  $v$ -inverter that maps an  $s \in \{0, 1\}^\ell$  to one of its pre-images chosen uniformly at random. Let  $(\text{SA-ECCenc}, \text{SA-ECCdec})$  be a stochastic affine-erasure correcting code with encoder  $\text{SA-ECCenc} : \{0, 1\}^n \times \mathcal{R}_2 \rightarrow \{0, 1\}^N$  that tolerates  $N - r$  erasures and decodes with success probability at least  $1 - \delta$ . Then the  $(\text{Share}, \text{Recst})$  defined as follows is an adaptive  $(\varepsilon + v, \delta)$ -SSS with threshold pair  $(t, r)$ .*

$$\begin{cases} \text{Share}(s) &= \text{SA-ECCenc}(\text{AExt}^{-1}(s)); \\ \text{Recst}(\tilde{v}) &= \text{AExt}(\text{SA-ECCdec}(\tilde{v})), \end{cases}$$

where  $\tilde{v}$  denotes an incomplete version of a share vector  $v \in \{0, 1\}^N$  with some of its components replaced by erasure symbols.

**Proof.** The  $(r, \delta)$ -reconstructability follows directly from the erasure correcting capability of the SA-ECC. For any  $\tilde{\mathbf{v}}$  with at most  $N - r$  erasure symbols and the rest of its components consistent with a valid codeword  $\mathbf{v} \in \{0, 1\}^N$ , the SA-ECC decoder identifies the unique codeword  $\mathbf{v}$  with probability  $1 - \delta$  over the encoder randomness. The corresponding SA-ECC message of  $\mathbf{v}$  is then inputted to AExt and the original secret  $\mathbf{s}$  is reconstructed with the same probability.

We next prove the  $(t, \varepsilon)$ -privacy. Without loss of generality, we first assume the inverter of the affine extractor is perfect, namely,  $v = 0$ . When  $v$  is negligible but not equal to zero, the overall privacy error will increase slightly, but still remain negligible. For any  $r \in \mathcal{R}_2$ , the affine encoder of SA-ECC is characterised by a matrix  $G_r \in \{0, 1\}^{n \times N}$  and an offset  $\Delta_r$ . For  $n$  unknowns  $\mathbf{x} = (x_1, \dots, x_n)$ , we have

$$\text{SA-ECCenc}(\mathbf{x}) = \mathbf{x}G_r + \Delta_r = (\mathbf{x}G_1, \dots, \mathbf{x}G_N) + \Delta_r,$$

where  $G_i = (g_{1,i}, \dots, g_{n,i})^T$  (here the subscript “ $r$ ” is omitted to avoid double subscripts) denotes the  $i$ th column of  $G_r$ ,  $i = 1, \dots, N$ . This means that knowing a component  $c_i$  of the SA-ECC codeword is equivalent to obtaining a linear equation  $c_i \oplus \Delta_i = \mathbf{x}G_i = g_{1,i}x_1 + \dots + g_{n,i}x_n$  about the  $n$  unknowns  $x_1, \dots, x_n$ , where  $\Delta_i$  (again, the subscript “ $r$ ” is omitted) denotes the  $i$ th component of  $\Delta_r$ . Now, we investigate the distribution of  $\text{View}_{\mathcal{A}}^{\mathcal{O}_t(\text{Share}(\mathbf{s}))}$  for any secret  $\mathbf{s} \in \{0, 1\}^\ell$  by finding the probability  $\Pr[\text{View}_{\mathcal{A}}^{\mathcal{O}_t(\text{Share}(\mathbf{s}))} = \mathbf{w}]$  for arbitrary  $\mathbf{w}$ . We then have

$$\begin{aligned} \Pr[\text{View}_{\mathcal{A}}^{\mathcal{O}_t(\text{Share}(\mathbf{s}))} = \mathbf{w}] &= \Pr[\text{View}_{\mathcal{A}}^{\mathcal{O}_t(\text{SA-ECCenc}(\mathbf{X}))} = \mathbf{w} | \text{AExt}(\mathbf{X}) = \mathbf{s}] \\ &= \frac{\Pr[\text{AExt}(\mathbf{X}) = \mathbf{s} | \text{View}_{\mathcal{A}}^{\text{SA-ECCenc}(\mathbf{X})} = \mathbf{w}] \cdot \Pr[\text{View}_{\mathcal{A}}^{\text{SA-ECCenc}(\mathbf{X})} = \mathbf{w}]}{\Pr[\text{AExt}(\mathbf{X}) = \mathbf{s}]} \\ &\stackrel{(i)}{=} \frac{(1 \pm \frac{\varepsilon}{2})2^{-\ell} \cdot \Pr[\text{View}_{\mathcal{A}}^{\text{SA-ECCenc}(\mathbf{X})} = \mathbf{w}]}{\Pr[\text{AExt}(\mathbf{X}) = \mathbf{s}]} \\ &\stackrel{(ii)}{=} \frac{(1 \pm \frac{\varepsilon}{2})2^{-\ell} \cdot \frac{2^{n - \text{rank}(\mathcal{A})}}{2^n}}{2^{-\ell}} \\ &= (1 \pm \frac{\varepsilon}{2}) \cdot 2^{-\text{rank}(\mathcal{A})}, \end{aligned}$$

where notations  $\mathbf{X}$ ,  $\pm$ ,  $\text{rank}(\mathcal{A})$  and  $(i)$ ,  $(ii)$  are explained as follows. In above, we first use the fact that  $\Pr[\text{View}_{\mathcal{A}}^{\mathcal{O}_t(\text{Share}(\mathbf{s}))} = \mathbf{w}]$  can be seen as the probability of uniformly selecting  $\mathbf{X}$  from  $\{0, 1\}^n$ , with the condition that  $\text{AExt}(\mathbf{X}) = \mathbf{s}$ . This is true because the sets  $\text{AExt}^{-1}(\mathbf{s})$  for all  $\mathbf{s}$ , partition  $\{0, 1\}^n$  and the rest follows from Definition 6. The shorthand “ $y = 1 \pm \frac{\varepsilon}{2}$ ” denotes “ $1 - \frac{\varepsilon}{2} \leq y \leq 1 + \frac{\varepsilon}{2}$ ”. The shorthand “ $\text{rank}(\mathcal{A})$ ” denotes the rank of the up to  $t$  columns of  $G$  corresponding to the index set  $\mathcal{A}$  adaptively chosen by  $\mathcal{A}$ . The equality  $(i)$  follows from the fact that AExt is an  $(n - t, 2^{-(\ell+1)}\varepsilon)$ -affine extractor and the uniform  $\mathbf{X}$  conditioned on at most  $t$  linear equations is an affine source with at least  $n - t$  bits entropy. The equality  $(ii)$  holds if and only if  $\mathbf{w}$  is in the set  $\{\text{SA-ECCenc}(\mathbf{x})_{\mathcal{A}} : \mathbf{x} \in \{0, 1\}^n\}$ . Indeed, consider  $\mathbf{X}$  as unknowns for equations, the number of solutions to the linear system  $\text{SA-ECCenc}(\mathbf{X})_{\mathcal{A}} = \mathbf{w}$  is either 0 or equal to  $2^{n - \text{rank}(\mathcal{A})}$ .

The distribution of  $\text{View}_{\mathcal{A}}^{\mathcal{O}_t(\text{Share}(\mathbf{s}))}$  for any secret  $\mathbf{s}$  is determined by the quantity  $\text{rank}(\mathcal{A})$ , which is independent of the secret  $\mathbf{s}$ . Let  $\mathbf{W}$  be the uniform distribution over the set  $\{\text{SA-ECCenc}(\mathbf{x})_{\mathcal{A}} : \mathbf{x} \in \{0, 1\}^n\}$ . Then by the triangular inequality, we have

$$\begin{aligned} &\text{SD} \left( \text{View}_{\mathcal{A}}^{\mathcal{O}_t(\text{Share}(\mathbf{s}_0))}; \text{View}_{\mathcal{A}}^{\mathcal{O}_t(\text{Share}(\mathbf{s}_1))} \right) \\ &\leq \text{SD} \left( \text{View}_{\mathcal{A}}^{\mathcal{O}_t(\text{Share}(\mathbf{s}_0))}; \mathbf{W} \right) + \text{SD} \left( \mathbf{W}; \text{View}_{\mathcal{A}}^{\mathcal{O}_t(\text{Share}(\mathbf{s}_1))} \right) \\ &\leq \frac{\varepsilon}{2} + \frac{\varepsilon}{2} = \varepsilon. \end{aligned}$$

When the inverter of the affine extractor is not perfect, the privacy error is upper bounded by  $\varepsilon + v$ . This concludes the privacy proof. ◀

There are explicit constructions of binary affine extractors that, given a constant fraction of entropy, outputs a constant fraction of random bits with exponentially small error (see Lemma 5). There are known methods for constructing an invertible affine extractor  $\text{AExt}'$  from any affine extractor  $\text{AExt}$  such that the constant fraction output size and exponentially small error properties are preserved. A simple method is to let  $\text{AExt}'(U_n||M) := \text{AExt}(U_n) \oplus M$  (see Appendix A for a discussion). This is summarized in the lemma below.

► **Lemma 20.** *For any  $\delta \in (0, 1]$ , there is an explicit seedless  $(\delta n, \varepsilon)$ -almost perfect affine extractor  $\text{AExt}: \{0, 1\}^n \rightarrow \{0, 1\}^m$  where  $m = \Omega(n)$  and  $\varepsilon = \exp(-\Omega(n))$ . Moreover, there is an efficiently computable  $\varepsilon$ -inverter for the extractor.*

**Proof.** Let  $\text{AExt}: \{0, 1\}^n \rightarrow \{0, 1\}^m$  be Bourgain’s affine extractor (Lemma 5) for entropy rate  $\mu$ , output length  $m = \Omega(n)$ , and achieving exponentially small error  $\varepsilon = \exp(-\Omega(n))$ . Using the one-time pad trick (Appendix A), we construct an invertible variant achieving output length  $m' = \Omega(m) = \Omega(n)$  and exponentially small error. Finally, we simply truncate the output length of the resulting extractor to  $m'' = \Omega(m') = \Omega(n)$  bits so that the closeness to uniformity measured by  $\ell_\infty$  norm required for almost-perfect extraction is satisfied. The truncated extractor is still invertible since the inverter can simply pad the given input with random bits and invoke the original inverter function. ◀

It now suffices to instantiate Theorem 19 with the explicit construction of SA-ECC and the invertible affine extractor  $\text{AExt}$  of Lemma 20. Let  $R_{ECC}$  denote the rate of the SA-ECC. Then we have  $R_{ECC} = \frac{n}{N}$ , where  $n$  is the input length of the affine extractor  $\text{AExt}$  and  $N$  is the number of players. The intuition of the construction in Theorem 19 is that if a uniform secret is shared and conditioning on the revealed shares the secret still has a uniform distribution (being the output of a randomness extractor), then no information is leaked. In fact, the proof of Theorem 19 above is this intuition made exact, with special care on handling the imperfectness of the affine extractor. So as long as the “source” of the affine extractor  $\text{AExt}$  has enough entropy, privacy is guaranteed. Here the “source” is the distribution  $U_n$  conditioned on the adversary’s view, which is the output of a  $t$ -bit affine function. The “source” then is affine and has at least  $n - \tau N = n(1 - \frac{\tau}{R_{ECC}})$  bits of entropy. Now as long as  $\tau < R_{ECC}$ , using the  $\text{AExt}$  from Lemma 5 (more precisely, an invertible affine extractor  $\text{AExt}' : \{0, 1\}^{n'} \rightarrow \{0, 1\}^\ell$  constructed from  $\text{AExt}$ ) with  $\mu = 1 - \frac{\tau}{R_{ECC}}$ , a constant fraction of random bits can be extracted with exponentially small error. This says that privacy is guaranteed for  $\tau \in [0, R_{ECC})$ . The stochastic affine ECC in Lemma 8 asymptotically achieves the rate  $1 - (1 - \rho) = \rho$ . We then have the following corollary.

► **Corollary 21.** *For any  $0 \leq \tau < \rho \leq 1$ , there is an explicit constant coding rate adaptive  $(\varepsilon, \delta)$ -SSS with relative threshold pair  $(\tau, \rho)$ , where  $\varepsilon$  and  $\delta$  are both negligible.*

The construction above achieves a constant coding rate for any  $(\tau, \rho)$  pair satisfying  $0 \leq \tau < \rho \leq 1$ . However, since the binary affine extractor in Lemma 5 does not extract all the entropy from the source and moreover the step that transforms an affine extractor into an invertible affine extractor incurs non-negligible overhead, the coding rate of the above construction does not approach  $\rho - \tau$ . We leave explicit constructions of binary SSS against adaptive adversary with better coding rate as an interesting technical open question.

## 6 Conclusion

We studied the problem of sharing *arbitrary long secrets* using constant length shares and required a nearly-threshold access structure. By nearly-threshold we mean a ramp scheme with arbitrarily small gap to number of players ratio. We show that by replacing perfect privacy and reconstructibility with slightly relaxed notions and inline with similar strong cryptographic notions, one can explicitly construct such nearly-threshold schemes. We gave two constructions with security against non-adaptive and adaptive adversaries, respectively, and proved optimality of the former. Our work also make a new connection between secret sharing and wiretap coding.

We presented our model and constructions for the extremal case of binary shares. However, we point out that the model and our constructions can be extended to shares over any desired alphabet size  $q$ . Using straightforward observations (such as assigning multiple shares to each player), this task reduces to extending the constructions over any prime  $q$ . In this case, the building blocks that we use; namely, the stochastic error-correcting code, seeded and seedless affine extractors need to be extended to the  $q$ -ary alphabet. The constructions [24, 19, 15] that we use, however, can be extended to general alphabets with straightforward modifications. The only exception is Bourgain's seedless affine extractor [7]. The extension of [7] to arbitrary alphabets is not straightforward and has been accomplished in a work by Yehudayoff [26].

Our constructions are not linear: even the explicit non-adaptive construction that uses an affine function for every fixing of the encoder's randomness does not result in a linear secret sharing. Linearity is an essential property in applications such as multiparty computation and so explicit constructions of *linear* secret sharing schemes in our model will be an important achievement. Yet another important direction for future work is deriving bounds and constructing optimal codes for finite length ( $N$ ) case. Such result will also be of high interest for wiretap coding.

---

## References

- 1 Vaneet Aggarwal, Lifeng Lai, A. Robert Calderbank, and H. Vincent Poor. Wiretap channel type II with an active eavesdropper. *IEEE International Symposium on Information Theory, ISIT 2009*, pages 1944–1948, 2009. doi:10.1109/ISIT.2009.5205631.
- 2 Mihir Bellare, Stefano Tessaro, and Alexander Vardy. Semantic Security for the Wiretap Channel. In *Advances in Cryptology - CRYPTO 2012*, volume 7417 of *Lecture Notes in Computer Science*, pages 294–311. Springer, 2012.
- 3 George R. Blakley. Safeguarding cryptographic keys. In *Proceedings of the 1979 AFIPS National Computer Conference*, pages 313–317, 1979.
- 4 Andrej Bogdanov, Siyao Guo, and Ilan Komargodski. Threshold Secret Sharing Requires a Linear Size Alphabet. In *Theory of Cryptography - TCC 2016-B*, pages 471–484, 2016.
- 5 Andrej Bogdanov, Yuval Ishai, Emanuele Viola, and Christopher Williamson. Bounded Indistinguishability and the Complexity of Recovering Secrets. In *Advances in Cryptology - CRYPTO*, pages 593–618, 2016.
- 6 Andrej Bogdanov and Christopher Williamson. Approximate Bounded Indistinguishability. In *International Colloquium on Automata, Languages, and Programming, ICALP*, pages 53:1–53:11, 2017.
- 7 Jean Bourgain. On the construction of affine extractors. *Geometric and Functional Analysis*, 17(1):33–57, 2007.



- 8 Ignacio Cascudo Pueyo, Ronald Cramer, and Chaoping Xing. Bounds on the Threshold Gap in Secret Sharing and its Applications. *IEEE Trans. Information Theory*, 59(9):5600–5612, 2013. doi:10.1109/TIT.2013.2264504.
- 9 Kuan Cheng, Yuval Ishai, and Xin Li. Near-Optimal Secret Sharing and Error Correcting Codes in AC0. In *Theory of Cryptography - TCC 2017, Part II*, pages 424–458, 2017.
- 10 Mahdi Cheraghchi. Nearly Optimal Robust Secret Sharing. *Designs, Codes and Cryptography*, pages 1–20, 2018.
- 11 Mahdi Cheraghchi, Frédéric Didier, and Amin Shokrollahi. Invertible Extractors and Wiretap Protocols. *IEEE Trans. Information Theory*, 58(2):1254–1274, 2012. doi:10.1109/TIT.2011.2170660.
- 12 Ronald Cramer, Ivan Damgård, and Jesper Buus Nielsen. *Secure Multiparty Computation and Secret Sharing*. Cambridge University Press, 2015. URL: <http://www.cambridge.org/de/academic/subjects/computer-science/cryptography-cryptology-and-coding/secure-multiparty-computation-and-secret-sharing?format=HB&isbn=9781107043053>.
- 13 Imre Csiszár and Janos Körner. Broadcast channels with confidential messages. *IEEE Trans. Information Theory*, 24(3):339–348, 1978. doi:10.1109/TIT.1978.1055892.
- 14 Venkatesan Guruswami. List decoding from erasures: bounds and code constructions. *IEEE Trans. Information Theory*, 49(11), 2003.
- 15 Venkatesan Guruswami and Adam D. Smith. Optimal Rate Code Constructions for Computationally Simple Channels. *J. ACM*, 63(4):35:1–35:37, 2016. doi:10.1109/FOCS.2010.74.
- 16 Xin Li. A New Approach to Affine Extractors and Dispersers. *IEEE Conference on Computational Complexity, CCC 2011*, pages 137–147, 2011.
- 17 Wakaha Ogata, Kaoru Kurosawa, and Shigeo Tsujii. Nonperfect Secret Sharing Schemes. *Advances in Cryptology - AUSCRYPT '92*, pages 56–66, 1992.
- 18 Lawrence H. Ozarow and Aaron D. Wyner. Wire-tap channel II. *The Bell System Technical Journal*, 63:2135–2157, December 1984.
- 19 Ran Raz, Omer Reingold, and Salil P. Vadhan. Extracting all the Randomness and Reducing the Error in Trevisan’s Extractors. *J. Comput. Syst. Sci.*, 65(1):97–128, 2002.
- 20 Adi Shamir. How to Share a Secret. *Commun. ACM*, 22(11):612–613, 1979.
- 21 Setareh Sharifian, Fuchun Lin, and Reihaneh Safavi-Naini. Hash-then-Encode: A Modular Semantically Secure Wiretap Code. In *Proceedings of the 2nd Workshop on Communication Security, WCS 2017*, pages 49–63. Lecture Notes in Electrical Engineering, 2017.
- 22 Adam D. Smith. Scrambling Adversarial Errors Using Few Random Bits, Optimal Information Reconciliation, and Better Private Codes. In *ACM-SIAM Symposium on Discrete Algorithms SODA '07*, pages 395–404. Society for Industrial and Applied Mathematics, 2007. URL: <http://dl.acm.org/citation.cfm?id=1283383.1283425>.
- 23 Douglas R. Stinson. An Explication of Secret Sharing Schemes. *Des. Codes Cryptography*, 2(4):357–390, 1992.
- 24 Luca Trevisan. Extractors and pseudorandom generators. *J. ACM*, 48(4):860–879, 2001.
- 25 Aaron D. Wyner. The wire-tap channel. *The Bell System Technical Journal*, 54(8):1355–1387, October 1975.
- 26 Amir Yehudayoff. Affine extractors over prime fields. *Combinatorica*, 31(2):245, 2011. doi:10.1007/s00493-011-2604-9.



### A One-Time-Pad trick of inverting extractors

The following trick to transform an efficient function into one that is also efficiently invertible has appeared in different form before (see for example [21]). We prove it here for the case of affine extractors, for completeness.

► **Lemma 22.** *Let  $\text{AExt}: \{0, 1\}^n \rightarrow \{0, 1\}^m$  be an affine  $(n, k)$ -extractor with error  $\varepsilon$ . Then  $\text{AExt}' : \{0, 1\}^{n+m} \rightarrow \{0, 1\}^m$  defined as follows is a  $\varepsilon$ -invertible affine  $(n+m, k+m)$ -extractor with error  $\varepsilon$ .*

$$\text{AExt}'(z) = \text{AExt}'(x||y) = \text{AExt}(x) + y,$$

where the input  $z \in \{0, 1\}^{n+m}$  is separated into two parts:  $x \in \{0, 1\}^n$  and  $y \in \{0, 1\}^m$ .

**Proof.** Let  $Z$  be a random variable with flat distribution supported on an affine subspace of  $\{0, 1\}^{n+m}$  of dimension at least  $k+m$ . Separate  $Z$  into two parts  $Z = (X||Y)$ , where  $X \in \{0, 1\}^n$  and  $Y \in \{0, 1\}^m$ . Then conditioned on any  $Y = y$ ,  $X$  has a distribution supported on an affine subspace of  $\{0, 1\}^n$  of dimension at least  $k$ . This asserts that conditioned on any  $Y = y$ ,

$$\text{SD}(\text{AExt}(X) + y; U_{\{0,1\}^m}) \leq \varepsilon.$$

Averaging over the distribution of  $Y$  concludes the extractor proof.

We next show an efficient inverter  $\text{AExt}'^{-1}$  for  $\text{AExt}'$ . For any  $s \in \{0, 1\}^m$ , define

$$\text{AExt}'^{-1}(s) = (U_n || \text{AExt}(U_n) + s).$$

The randomised function  $\text{AExt}'^{-1}$  is efficient and  $\text{AExt}'^{-1}(U_m) \stackrel{\varepsilon}{\sim} U_{n+m}$ . ◀

# On the Communication Complexity of High-Dimensional Permutations

Nati Linial<sup>1</sup>

Hebrew University of Jerusalem, Jerusalem, Israel  
nati@cs.huji.ac.i

Toniann Pitassi

University of Toronto, Toronto, Canada and IAS, Princeton, U.S.A.  
toni@cs.toronto.edu

Adi Shraibman

The Academic College of Tel-Aviv-Yaffo, Tel-Aviv, Israel  
adish@mta.ac.il

---

## Abstract

We study the multiparty communication complexity of high dimensional permutations in the Number On the Forehead (NOF) model. This model is due to Chandra, Furst and Lipton (CFL) who also gave a nontrivial protocol for the Exactly- $n$  problem where three players receive integer inputs and need to decide if their inputs sum to a given integer  $n$ . There is a considerable body of literature dealing with the same problem, where  $(\mathbb{N}, +)$  is replaced by some other abelian group. Our work can be viewed as a far-reaching extension of this line of research. We show that the known lower bounds for that group-theoretic problem apply to all high dimensional permutations. We introduce new proof techniques that reveal new and unexpected connections between NOF communication complexity of permutations and a variety of well-known problems in combinatorics. We also give a direct *algorithmic* protocol for Exactly- $n$ . In contrast, all previous constructions relied on large sets of integers without a 3-term arithmetic progression.

**2012 ACM Subject Classification** Theory of computation → Communication complexity

**Keywords and phrases** High dimensional permutations, Number On the Forehead model, Additive combinatorics

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2019.54

**Related Version** A full version of the paper is available at <https://arxiv.org/abs/1706.02207>.

## 1 Introduction

The multiplayer Number On the Forehead (NOF) model of communication complexity was first introduced by Chandra, Furst and Lipton [13]. Here  $k$  players need to evaluate a given function  $f : [n]^k \rightarrow \{0, 1\}$ , where we think of  $f$  as having  $k$  arguments  $x_1, \dots, x_k$ , each comprised of  $\log n$  bits. The  $i$ -th input vector  $x_i$  is placed metaphorically on player  $i$ 's forehead, so that every player sees the whole input but one argument. Players communicate by writing bits on a shared blackboard in order to compute  $f$ .

The NOF communication model has turned out to be a fascinating, though exceedingly difficult object of study. Indeed, good lower bounds in the NOF model would resolve several longstanding open problems in complexity theory, such as lower bounds on the size of  $ACC^0$

---

<sup>1</sup> Supported in part by ERC grant 339096, High-dimensional combinatorics.



circuits for a natural function in  $P$  [40, 23]. They also imply lower bounds for branching programs, time-space tradeoffs for Turing machines [26], and proof complexity lower bounds [8]. The implications of good NOF lower bounds go in other, less expected directions as well. E.g., knowing the communication complexity of specific natural functions, even for  $k = 3$ , would have profound implications in graph theory and combinatorics. Finally, the search for nontrivial protocols in this area is a wonderful challenge for algorithms designers. There is a short list of such beautiful examples [13, 22] which begs to be extended.

Furthermore, our understanding of NOF communication complexity, even for  $k = 3$  players, lags well behind our understanding of the standard model ( $k = 2$  players). This gap is usually attributed to the dearth of proof techniques in the NOF setting. In the 2-party setting, many measures of complexity allow us to prove both upper and lower bounds. Such measures include matrix rank, various matrix norms, nonnegative rank, discrepancy, corruption bounds and information complexity. Most of these measures are computationally simple and admit dual characterizations which are very helpful in proving both upper and lower bounds. On the other hand, in the NOF setting for  $k \geq 3$ , the key combinatorial objects are cylinder intersections (rather than combinatorial rectangles) and tensor norms. These are much more complex, and thus far have resisted a workable characterization.

A case in point is the separation of randomized from deterministic communication complexity. The 2-party *equality function* has a randomized protocol of bounded cost, whereas a simple rank argument shows that every deterministic protocol must incur linear cost. This provides an optimal separation of deterministic and randomized communication complexity [26]. On the other hand, for  $k \geq 3$ , the best *explicit* separation between nondeterministic and randomized NOF complexity is logarithmic, even though counting arguments yield linear separations [7]. The Exactly- $n$  function is defined as follows: Input  $x_1, \dots, x_k \in [n]$  is accepted iff  $\sum_i x_i = n$ . In their seminal paper, Chandra, Furst and Lipton [13] conjectured that Exactly- $n$  achieves a strong separation, and also connected the communication complexity of this function to well-known problems in additive combinatorics. But thus far, despite considerable research effort, the lower bounds for Exactly- $n$  are much weaker even than the best (logarithmic) explicit separations.

The main goal of our work is to further investigate the connections between NOF complexity of functions and questions in additive combinatorics, with the hope of stimulating further research to make progress in both directions. A large and rapidly growing body of work establishes interesting relationships between problems in additive combinatorics and complexity theory. For example, the study of expander graphs and extractors, pseudorandomness, and property testing is closely related, some time even synonymous with similar notions in additive combinatorics. Moreover, techniques from complexity theory have been useful in additive combinatorics and vice versa. Some recent examples include the proof of the cap-set conjecture [14, 17] and Dvir's resolution [15] of the finite field Kakeya problem, as well as the beautiful interplay between dense model theorems in additive combinatorics and the notions of boosting and hardcore sets from complexity theory [11, 38, 29].

Here we consider a broad class of functions called high dimensional permutations. We uncover strong connections between the NOF communication complexity of these functions and several fundamental problems in additive combinatorics. Originally defined in [27], a  $(k - 1)$ -dimensional permutation is a function  $f : [n]^k \rightarrow \{0, 1\}$  such that for every index  $k \geq i \geq 1$  and for every choice of  $x_1, \dots, x_{i-1}, x_{i+1}, \dots, x_k \in [n]$ , there is exactly one value of  $x_i \in [n]$  for which  $f(x_1, \dots, x_{i-1}, x_i, x_{i+1}, \dots, x_k) = 1$ . This class of functions generalizes many well-studied functions in communication complexity. It is also closely related to many other functions such as the Exactly- $n$  function mentioned above.

We will show that many well-studied problems in NOF complexity are not just related, but are in fact identical or nearly identical to central problems in additive combinatorics. We feel that this mutual relation deserves much more attention, and that progress in this area is likely to greatly advance both domains. Specifically we believe that the study of the communication complexity of high dimensional permutations and related graph functions (defined in [7]) is a worthwhile undertaking that will help us develop new lower bounds proof techniques for the notoriously difficult NOF model. Using these connections, we make modest progress on several upper and lower bounds in NOF communication complexity.

## 1.1 Our Contributions

As mentioned above, our main goal and contribution is to unveil the strong relationships between the NOF complexity of high dimensional permutation problems and central problems in additive combinatorics and Ramsey theory. Already the founding paper of Chandra, Furst and Lipton [13] makes a connection between the NOF complexity of Exactly- $n$  and the areas of Ramsey theory and additive combinatorics. A more general framework was introduced in [10]: Given an abelian group  $G$  and  $T \in G$ , the function  $f_{k,T}^G$  evaluates to 1 on input  $x_1, \dots, x_k \in G$  iff  $\sum_i x_i = T$  (this expression is well-defined since  $G$  is abelian). The functions  $f_{k,T}^G$  are high dimensional permutations. (Note that this holds as well for non-abelian  $G$ , though we need to specify the order at which  $\prod_i x_i$  is evaluated). Another strong connection is that the Hales-Jewett theorem, a cornerstone of Ramsey theory, can be interpreted in terms of communication complexity [35].

We establish a new and close connection between the NOF communication complexity of high dimensional permutations and dense Ruzsa-Szemerédi graphs. These graphs appear in various contexts in Combinatorics, Computer Science and Information Theory, thus highlighting new connections between communication complexity and these various problems. For example, an efficient deterministic communication protocol for any permutation yields an efficient wiring scheme for shared directional multi-channels. For more on this, see e.g., [12] and [4]. In the classical,  $k = 2$  case, monochromatic submatrices play a key role in the theory. For higher  $k$  this is replaced by the much more poorly understood monochromatic *cylinder intersection*. Naturally, much of our work here revolves around these complicated objects. However, in certain simple cases we are able to get a grip on the largest size of a cylinder intersection that contains only 1-inputs of  $f$ . As we show, in this case knowledge of this quantity essentially determines the NOF communication complexity of  $f$  (see more on this in the next section). The case in question is  $k = 3$  and the group  $G = \mathbb{Z}_2^n$ . As we show, the size of the largest cylinder intersection containing only 1-inputs of  $f$  is the largest cardinality of a subset  $W \subseteq \mathbb{Z}_4^n$  such that for every three distinct members  $\mathbf{x}, \mathbf{y}, \mathbf{z} \in W$  there is an index  $1 \leq i \leq n$  for which  $(x_i, y_i, z_i) \notin X$ , where

$$X = \{(0, 0, 0), (1, 1, 1), (2, 2, 2), (3, 3, 3), (0, 1, 2), (1, 0, 3), (2, 3, 0), (3, 2, 1)\}.$$

This parameter may seem artificial, but in fact, this framework includes several important problems in combinatorics, for different choices of  $X$ . Thus, if we take  $X := \{(0, 0, 0), (1, 1, 1), (2, 2, 2), (3, 3, 3), (0, 1, 2)\}$ , then this becomes precisely the density Hales-Jewett problem, solved in [19]. Also, if  $X$  is comprised of all triplets  $(a, b, c) \in \mathbb{Z}_4^3$  with  $a + c = 2b$ , we arrive at the cap-set problem for  $\mathbb{Z}_4^n$  which was recently settled in breakthrough papers by Croot, Lev, and Pach, and by Ellenberg and Gijswijt [14, 17]. In the next subsection we list our new results that stem from these connections.

### 1.1.1 Upper Bounds

We give a new algorithm for Exactly- $n$  as well as several other instances of  $f_{k,T}^G$ . All previous upper bounds for these functions crucially depend on Behrend's famous construction [9] of a large set of integers with no 3-term arithmetic progressions. This yields a large monochromatic cylinder intersection, and a simple probabilistic translation lemma then shows how to cover the whole space by monochromatic cylinder intersections, thus providing an efficient protocol. To show that this indeed yields a large monochromatic cylinder intersection, we appeal to the notion of *corner-free* sets [2, 36] which here, too, plays a key role. We cannot realistically hope to improve the bounds by finding a construction better than Behrend's, in view of the many such failed attempts throughout the past 70 years (but note [16]). However, Behrend's construction is actually more than we need. The solution of  $f_{k,T}^G$  only requires corner-free sets. That is, 3-term AP freeness implies corner-freeness, but we do not expect that the two concepts are equivalent. We take a first step in this direction and give a new algorithm which is not dependent on 3-term AP freeness. We hope that this indicates a viable approach to improved protocols for the Exactly- $n$  function. We also describe a nontrivial protocol for the  $f_{3,T}^G$  problem for  $G = \mathbb{Z}_2^n$ .

### 1.1.2 Lower Bounds

We give a counting argument which shows that almost every  $k$ -dimensional permutation has communication complexity  $\Omega(\frac{\log n}{k})$ . Clearly, up to the  $\frac{1}{k}$  factor, this is as high as this quantity can get. Our proof relies on a recent lower bound of Keevash [25] on the number of high-dimensional permutations. This method resembles the counting argument for graph functions of [7], which does not apply, though, to permutations.

Regarding bounds on explicit functions, we prove a weak upper bound on the size of a 1-monochromatic cylinder intersection for any permutation (in fact our result holds for a wider family of functions that we call linjections). This bound uses a graph theoretic characterization of the communication complexity of permutations, connecting it also to Ruzsa-Szemerédi graphs. Not unexpectedly, our proof mirrors a similar result for Ruzsa-Szemerédi graphs: Solymosi [36] showed that the multidimensional Szemerédi theorem follows from the triangle removal lemma. We adapt Solymosi's proof to our context. The main tools in the proof are thus the graph and the hypergraph removal lemmas.

We note that previous results were limited to the  $f_{k,T}^G$  function for abelian groups with many factors, whereas ours works for general permutations. To emphasize the significance of the last point, consider the NOF complexity of following three classes of functions: (i) Permutations that come from Abelian groups, (ii) Those that come from general groups, (iii) Latin squares. We consider each such function up to an arbitrary renaming of rows and columns. The sizes of these three classes differ very substantially. For a given order  $n$  the size of the relevant class is (i)  $\exp(O(\sqrt{\log n}))$ , (ii) At most  $\exp((\frac{2}{27} + o(1)) \log^3 n)$ , and (iii)  $((1 + o(1)) \frac{n}{e^2})^{n^2}$ .

For  $k = 3$  we can say more: The communication complexity of every 2-dimensional permutation  $[n]^3 \rightarrow \{0, 1\}$  is  $\Omega(\log \log \log n)$ . This extends the lower bound of [10] from the realm of abelian groups to all permutations. The proof of the this lower bound uses only elementary counting arguments, and is closely related to the result of [20] on monochromatic corners on the integer grid.

The above lower bound also implies a result of Meshulam that was derived toward the study of shared directional multi-channels. Meshulam's result appears as Proposition 4.3 in [4], where further background can be found.

## 1.2 Related Work

The NOF model was introduced by Chandra, Furst and Lipton in [13]. One of the functions they consider is Exactly- $n : [n]^3 \rightarrow \{0, 1\}$ . For  $x, y, z \in [n]$ , we let Exactly- $n(x, y, z) = 1$  if and only if  $x + y + z = n$ . Surprisingly, they proved that the communication complexity of this function is only  $O(\sqrt{\log n})$ , but their proof yields no explicit protocol. Although this function is not a permutation, the proofs go through as well for the mod  $n$  permutation [10], and thus far this is the most efficient protocol found for any permutation. The protocol of [13] is based on Behrend's famous construction [9] of a large subset of  $[n]$  with no three-term arithmetic progression. In addition, they prove an inexplicit lower bound of  $\omega_n(1)$  on the complexity of Exactly- $n$ . This is based on Gallai's result [21, p. 38] that every finite coloring of a Euclidean space contains a monochromatic homothet of every finite set in that space.

Beigel, Gasarch and Glenn [10] considered the more general  $f_{k,T}^G$  problem, where  $G$  is an abelian group,  $T$  is an element of  $G$  and  $k \geq 2$  an integer. In this scenario  $k$  players need to decide whether  $x_1 + x_2 + \dots + x_k = T$ . They show that the communication complexity of  $f_{3,T}^G$  is at least  $\Omega(\log \log \log n)$  for every abelian group  $G$  and any  $T \in G$ . For the case  $G = \mathbb{Z}_n$ , this follows as well from [20] and a recent result of Shkerdov [34] also yields a similar lower bound for every abelian group  $G$ . For general  $k \geq 3$  and for an abelian group  $G$  that is the product of  $t$  cyclic groups, it is shown in [10] that the deterministic NOF complexity of  $f_{k,T}^G$  is  $\omega_t(1)$ . The proof is by reduction to a lower bound from [37], that is based on the Hales-Jewett Theorem (see [21]).

Note that  $f_{k,T}^G$  can be defined as well in non-abelian groups  $G$ . Namely,  $f_{k,T}^G(x_1, \dots, x_k) = 1$  iff  $x_1 \cdot x_2 \cdot \dots \cdot x_k = T$ , where now the order of multiplication matters. Note also that the function  $f_{k,T}^G$  is a permutation for every group  $G$ , every  $T \in G$  and  $k \geq 2$ .

As mentioned above, [7] studies graph functions and give a nonexplicit strong separation between randomized and deterministic NOF complexity. To be precise, this counting argument shows that most graph functions  $f : [n]^{k-1} \times [N] \rightarrow \{0, 1\}$  with  $N \cong \sqrt{\frac{n}{k}}$  have deterministic communication complexity  $\Omega(\log \frac{n}{k})$ . Still, even for  $k = 3$  it remains open to find *explicit* graph functions with high deterministic communication complexity. Currently, the best lower bound on the deterministic communication complexity of a graph function  $f : [n]^{k-1} \times [N] \rightarrow \{0, 1\}$  for  $k \geq 3$  is  $\Omega(\log \log n)$  proved in [7] (using also results from [5]). We note that the discrepancy method, used to establish NOF lower bounds (e.g., [6]), cannot be utilized here since it also applies to randomized communication complexity.

Lastly, we comment on the Hales-Jewett theorem, a pillar of Ramsey theory. It was previously applied in the study of the combinatorial problems mentioned above. It turns out that this theorem has an equivalent formulation in the language of communication complexity [35], and is tightly coupled with the NOF multiparty communication complexity of high dimensional permutations.

## 2 Basics

### 2.1 NOF Communication Complexity

In the Number On the Forehead (NOF) multiparty communication complexity game,  $k$  players collaborate to compute a function  $f : X_1 \times \dots \times X_k \rightarrow \{0, 1\}$ . Usually,  $X_i = [n]$  for all  $i \in [k]$ , but we also consider occasionally a variation where the last player is exceptional and  $X_k = [N]$  for some integer  $N$  that is not necessarily equal to  $n$ .

For  $(x_1, \dots, x_k) \in X_1 \times \dots \times X_k$ , and for each  $i \in [k]$ , player  $i$  receives  $x^{-i} \in X_1 \times \dots \times X_{i-1} \times X_{i+1} \times \dots \times X_k$ ; that is, all but  $x_i$ . The players exchange bits according to

an agreed-upon protocol, by writing them on a publicly visible blackboard. The protocol specifies, for every possible blackboard contents, whether or not the communication is ongoing. It shows the final output when the communication is over, and shows the next player to speak if the communication is still ongoing. The protocol also specifies what each player writes as a function of the blackboard contents and of the inputs seen by that player. The *cost* of the protocol is the maximum number of bits written on the blackboard.

The *deterministic communication complexity* of  $f$ ,  $D_k(f)$ , is the minimum cost of a deterministic protocol for  $f$  that always outputs the correct answer. A randomized protocol of cost  $c$  is just a distribution over deterministic protocols each of cost at most  $c$ . For  $0 \leq \epsilon < 1/2$ , the *randomized communication complexity* of  $f$ ,  $R_{k,\epsilon}(f)$ , is the minimum cost over randomized protocols such that for every input, err with probability at most  $\epsilon$  (over the distribution of deterministic protocols).

In the  $k = 2$  players case, the key combinatorial objects of study are combinatorial rectangles: Every cost- $c$  communication protocol for  $f : X_1 \times X_2 \rightarrow \{0, 1\}$  partitions  $X_1 \times X_2$  into  $2^c$  monochromatic combinatorial rectangles. For  $k$ -party NOF communication, a cost- $c$  protocol induces a partition of  $X_1 \times \dots \times X_k$  into  $2^c$  *monochromatic cylinder intersections*:

► **Definition 1.** A *cylinder in dimension  $i$*  is a subset  $S \subseteq \prod X_i$  such that if  $(x_1, \dots, x_k) \in S$ , then  $(x_1, \dots, x_{i-1}, x'_i, x_{i+1}, \dots, x_k) \in S$  for all  $x'_i$ . A *cylinder intersection* is a set of the form  $\bigcap_{i=1}^k T_i$ , where  $T_i$  is a cylinder in dimension  $i$ .

## 2.2 Graph Functions, Permutations and Linjections

► **Definition 2.** The line  $L \subseteq [n]^k$ , defined by a pair  $(a, i)$ , where  $a \in [n]^{k-1}$ ,  $i \in [k]$ , is the set of vectors  $v \in [n]^k$  such that  $v^{-i} = a$  and  $v_i$  is an arbitrary element in  $[n]$ .

► **Definition 3.** A function  $f : [n]^{k-1} \times [N] \rightarrow \{0, 1\}$  is a *graph function* if for every  $(x_1, \dots, x_{k-1})$  there is a unique  $b \in [N]$  such that  $f(x_1, \dots, x_{k-1}, b) = 1$ . In other words, every line in the  $k^{\text{th}}$  dimension,  $L = (a, k)$ , intersects  $f^{-1}(1)$  in exactly one point.

Associated with every graph function  $f : [n]^{k-1} \times [N] \rightarrow \{0, 1\}$  is a map  $A(f) : [n]^{k-1} \rightarrow [N]$ , where  $A(f)(x_1, \dots, x_{k-1}) = y$  if and only if  $f(x_1, \dots, x_{k-1}, y) = 1$ . We consider the two as one and the same object and freely switch back and forth between the two descriptions.

► **Definition 4.** Let  $f : [n]^{k-1} \times [N] \rightarrow \{0, 1\}$  be a graph function. We denote by  $\alpha_k(f)$  the largest size of a cylinder intersection that is contained in  $f^{-1}(1)$ . In other words, the largest cardinality of 1-monochromatic cylinder intersection with respect to  $f$ . Also, let  $\chi_k(f)$  be the least number of 1-monochromatic cylinder intersections whose union is  $f^{-1}(1)$ . We omit the subscript  $k$  when it is clear from context.

Given a graph function  $f$ , the measure  $\chi(f)$  corresponds to the nondeterministic NOF communication complexity of  $f$ , since it is a covering of the 1's of  $f$  by cylinder intersections [26]. In general, the nondeterministic NOF communication complexity of a Boolean function can be much smaller than the deterministic complexity – in fact, for the set disjointness function, nondeterministic complexity is logarithmic in the deterministic complexity (for constant  $k$ ). However, graph functions are special; the following lemma shows that for graph functions, the two notions basically coincide. The proof is an adaptation of a proof from [13]; see also [10, 7] for similar arguments.

► **Theorem 5.** For every graph function  $f : [n]^{k-1} \times [N] \rightarrow \{0, 1\}$ ,  $\log \chi_k(f) \leq D_k(f) \leq \lceil \log \chi_k(f) \rceil + k - 1$ .



A communication protocol in which players write only one message on the board, of arbitrary length is called a *one-way* protocol. This applies to the protocol in the proof of Theorem 5. The restriction to a single message per player may make one-way protocols much weaker than standard protocols [30, 5]. However for graph functions, one-way protocols and regular protocols are equally powerful:

► **Corollary 6.** *For every graph function  $f : [n]^{k-1} \times [N] \rightarrow \{0, 1\}$ ,  $D_k(f) \leq D_k^1(f) \leq D_k(f) + k$  where  $D_k^1(f)$  is the one-way communication complexity of  $f$ .*

Implicit in the proofs of the above statements is the fact that for graph functions, monochromatic cylinder intersections can be nicely characterized by forbidden (dual) objects called *stars*, which we define next. We will see in the next section that stars are very closely connected to corners (and higher dimensional generalizations) in Ramsey theory.

► **Definition 7.** A *star*  $Star(\mathbf{x}, \mathbf{x}')$  is a subset of  $[n]^{k-1} \times [N]$  of the form

$$\{(x'_1, x_2, \dots, x_k), (x_1, x'_2, \dots, x_k), \dots, (x_1, x_2, \dots, x'_k)\},$$

where  $x_i \neq x'_i$  for each  $i$ . We refer to  $\mathbf{x} = (x_1, x_2, \dots, x_k)$  as the star's *center*, and note that the center does *not* belong to the star.

► **Lemma 8.** *Let  $f : [n]^{k-1} \times [N] \rightarrow \{0, 1\}$  be a graph function, and let  $S \subseteq f^{-1}(1)$ . Then  $S$  is a (1-monochromatic) cylinder intersection with respect to  $f$  if and only if it does not contain a star.*

Next we define high dimensional permutations and linjections.

► **Definition 9.** A  $(k-1)$ -dimensional permutation of order- $n$  is a map  $f : [n]^k \rightarrow \{0, 1\}$  with the property that for every line  $L = (a, i)$  in  $[n]^k$ ,  $|L \cap f^{-1}(1)| = 1$ .

In other words,  $f$  is a permutation function if and only if every line contains a unique 1 entry. This property is easily seen to be equivalent to the property that for every choice of  $x_1, \dots, x_{i-1}, x_{i+1}, \dots, x_k \in [n]$ , there is exactly one value,  $A_i(x^{-i})$  for  $x_i \in [n]$  such that  $f(x_1, \dots, x_{i-1}, A_i(x^{-i}), x_{i+1}, \dots, x_k) = 1$ .

► **Example 10.** For the sake of gaining better intuition we often consider the important special case  $k = 3$ . This is insightful, since 2-dimensional permutations  $f : [n]^3 \rightarrow \{0, 1\}$  are synonymous with Latin squares. In this case  $A(f)$  is an  $n \times n$  matrix with entries in  $[n]$  where every row and column contains each element in  $[n]$  exactly once. Here we see an elementary but important connection with additive combinatorics; stars coincide with the well-studied notion of *corners* [2, 36]. A star is a triplet of entries in  $f^{-1}(1)$ ,  $(x, y, z')$ ,  $(x', y, z)$ ,  $(x, y', z)$ , which corresponds to the “corner” or “ $A$ -star”,  $(x, y)$ ,  $(x', y)$ ,  $(x, y')$ , where  $A(x', y)$  and  $A(x, y')$  have the same value ( $z$ ), but  $A(x, y)$  has a different value  $z'$ .

► **Example 11.** High dimensional permutations generalize the family of functions  $f_{k,T}^G$  for abelian groups  $G$ . For this communication problem, each player receives (on his/her forehead) an element  $x_i \in G$ , and they want to decide whether or not  $x_1 + \dots + x_k = T$ , that is, whether the sum of the elements is exactly  $T$ .

We can further generalize the notion of a permutation function as follows.

► **Definition 12.** A *linjection* is a graph function  $f : [n]^{k-1} \times [N] \rightarrow \{0, 1\}$  with  $N \geq n$  such that  $|f^{-1}(1)| = n^{k-1}$  and every line contains **at most** one point at which  $f = 1$ . A function  $f$  is a linjection if and only if the restriction of  $A(f)$  to any line is an injection.

Linjections are graph functions where  $N \geq n$ , (with permutation functions corresponding to  $n = N$ ), but not vice versa. Determining the least possible communication complexity of a linjection in certain dimensions is an interesting and challenging problem. Henceforth, we use and study the following two notions.

► **Definition 13.** Define  $\alpha_k(n, N) = \max_f \alpha_k(f)$ , and  $\chi_k(n, N) = \min_f \chi_k(f)$ , both taken over all linjections  $f : [n]^{k-1} \times [N] \rightarrow \{0, 1\}$ .

Note that  $\chi_k(n, N) \geq n^{k-1}/\alpha_k(n, N)$ .

### 3 High Dimensional Permutations and Additive Combinatorics

#### 3.1 A Graph-theoretic Characterization

In this section we give a new characterization of  $\alpha_k$  which will turn out to be a variant of the maximum density of Ruzsa-Szemerédi graphs. We start with the case  $k = 3$ .

Recall that we can view a linjection  $f : [n]^2 \times [N] \rightarrow \{0, 1\}$  as an  $n \times n$  matrix,  $A = A(f)$  with entries from  $[N]$ . Alternatively we view it as a tripartite graph  $G(A)$  with parts  $R = [n]$ ,  $C = [n]$  and  $W \subseteq [N]$ . Its edge set is defined as follows: for every triple  $(x, y, b) \in f^{-1}(1)$ , we add the triangle  $(x, y), (y, b), (x, b)$ ,  $x \in R, y \in C, b \in W$  to  $G(A)$ . In particular,  $R \cup C$  span a complete bipartite subgraph of  $G(A)$  and  $(i, b), i \in R, b \in W$  is an edge iff there is a  $b$  entry in row  $i$  of  $A$ , likewise for columns.

Let us consider the triangles  $\langle x, y, b \rangle$ ,  $x \in R, y \in C, b \in W$ , in  $G(A)$ . A triangle  $\langle x, y, b \rangle$  in  $G$  is *trivial* if  $A(x, y) = b$ . However, there can also be nontrivial (induced) triangles in  $G$ , which correspond to centers of stars. We define a *G-star* to be a triple of triangles in  $G$  of the form  $\langle x, y, b' \rangle, \langle x', y, b \rangle, \langle x, y', b \rangle$ . The point is that while these (trivial) triangles are edge-disjoint, their union contains the additional induced triangle  $\langle x, y, b \rangle$ . Define  $\bar{\alpha}(G)$  to be the largest cardinality of a family of edge-disjoint triangles that contains no *G-star*. In other words, a family of edge-disjoint triangles the union of which contains no additional triangle.

Let  $\bar{\alpha}(n, N) = \max_G \bar{\alpha}(G)$  where the maximum is over subgraphs of  $K_{n,n,N}$ . Then:

► **Theorem 14.** For every two integers  $n, N > 0$ , if  $n \leq N$  then  $\alpha_3(n, N) \leq \bar{\alpha}(n, N)$ . If  $N \geq 2n - 1$ , then  $\alpha_3(n, N) = \bar{\alpha}(n, N)$ .

**Proof.** We show first that  $\alpha_3(n, N) \leq \bar{\alpha}(n, N)$ . Let  $f : [n] \times [n] \times [N] \rightarrow \{0, 1\}$  be a linjection and let  $S \subseteq [n] \times [n] \times [N]$  be a star-free subset of  $f^{-1}(1)$ . We prove the claim by constructing a *G-star-free* family  $T$  of  $|S|$  edge-disjoint triangles in  $G = G(A(f))$ . Let

$$T = \{\langle x, y, b \rangle \mid (x, y, b) \in S\}.$$

The claim follows, since stars  $\{(x', y, b), (x, y', b), (x, y, b)\}$  correspond to *G-stars* in  $T$ . Next we prove the reverse inequality  $\alpha_3(n, N) \geq \bar{\alpha}(n, N)$  when  $N \geq 2n - 1$ .

Given a *G-star-free* family  $T$  of edge-disjoint triangles in a subgraph  $G$  of  $K_{n,n,N}$ , we find a linjection  $A : [n] \times [n] \rightarrow [N]$  that contains an *A-star-free* subset  $S \subset [n]^2$  of size  $|T|$ . In the proof we actually first construct  $S$  and only then proceed to define  $A$  in full.

We define  $S$  to be the projection of  $T$  to its first two coordinates. Namely,

$$S = \{(x, y) \mid \langle x, y, b \rangle \in T \text{ for some } b\}.$$

To define  $A$ , we first let  $A(x, y) = b$  for every  $\langle x, y, b \rangle \in T$ .

Since  $T$  is  $G$ -star-free, it follows that  $S$  is  $A$ -star-free. What is missing is that  $A$  is only partially defined. We show that when  $N \geq 2n - 1$  this partial definition can be extended to a linjection. Since the triangles in  $T$  are edge-disjoint it follows that in the partially defined  $A$ , no value appears more than once in any row or column. It remains to define  $A$  on all the entries outside of  $S$  and maintain this property. Indeed this can be done entry by entry. At worst there are  $2n - 2$  values that are forbidden for the entry of  $A$  that we attempt to define next, and therefore there is always an acceptable choice. ◀

**General  $k$ .** The construction for general  $k$  is a natural extension of the case  $k = 3$ . We associate with every linjection  $A : [n]^{k-1} \rightarrow [N]$  a  $k$ -partite  $(k-1)$ -uniform hypergraph  $H(A)$ . The parts of the vertex set are denoted  $Q_1, \dots, Q_{k-1}$  and  $W$ . Each  $Q_i$  is a copy of  $[n]$  and, as above,  $W$  is the range of  $A$ . There is a complete  $(k-1)$ -partite hypergraph on the  $k-1$  parts  $Q_1, \dots, Q_{k-1}$ . Given  $x_1 \in Q_1, \dots, x_{i-1} \in Q_{i-1}, x_{i+1} \in Q_{i+1}, \dots, x_{k-1} \in Q_{k-1}$  and  $w \in W$ , we put the hyperedge  $x_1, \dots, x_{i-1}, x_{i+1}, \dots, x_{k-1}, w$  in  $H(A)$  iff there is a (necessarily unique)  $x_i^* \in [n]$  for which  $A(x_1, \dots, x_{i-1}, x_i^*, x_{i+1}, \dots, x_{k-1}) = w$ .

We proceed to investigate *cliques* in  $H(A)$ , i.e., sets of  $k$  vertices, every  $k-1$  of which form an edge. For  $k = 3$ , we distinguished between those triangles in  $G(A)$  that correspond to an entry in  $[n]^2$  and those that form a star, and a similar distinction applies for general  $k$ .

It is easy to see that if  $A(x_1, \dots, x_{k-1}) = w$ , then  $x_1, \dots, x_{k-1}, w$  form a clique. Such a clique is considered *trivial*. In contrast,  $x_1, \dots, x_{k-1}, w$  is a nontrivial clique iff for every  $i$  there exists an  $x'_i \neq x_i$  such that  $A(x_1, \dots, x_{i-1}, x'_i, x_{i+1}, \dots, x_{k-1}) = w$ .

As above, we define for  $H = H(A)$  the parameter  $\bar{\alpha}_k(H)$ . It is the largest size of a family  $K$  of cliques in  $H$  such that: (i) No two share a hyperedge, and (ii) The hypergraph comprised of all cliques in  $K$  contains no additional cliques. Let  $\bar{\alpha}_k(n, N) = \max_H \bar{\alpha}_k(H)$  over all  $k$ -partite  $(k-1)$ -uniform hypergraphs  $H$ . Then

▶ **Theorem 15.** *For every two integers  $n \leq N$ ,  $\alpha_k(n, N) \leq \bar{\alpha}_k(n, N)$ , and if  $N > (k-1)(n-1)$  then  $\alpha_k(n, N) = \bar{\alpha}_k(n, N)$ .*

The proof is similar to the proof of Theorem 14 and appears in the full paper.

As the proofs show,  $\bar{\alpha}_k(n, N)$  is the largest cardinality of a star-free subset of  $[n]^{k-1} \times [N]$  that meets every line in  $[n]^{k-1} \times [N]$  at most once. To qualify for  $\alpha_k(n, N)$  this subset must, in addition, be extendable to a linjection, so clearly  $\bar{\alpha}_k(n, N) \geq \alpha_k(n, N)$ . We wonder whether this additional requirement creates a substantial difference between the two parameters. Specifically, how are  $\bar{\alpha}_k(n, N)$  and  $\alpha_k(n, N)$  related in the range  $n \leq N \leq (k-1)(n-1)$ ? These two parameters need not be equal in this range, since  $\alpha_3(4, 4) = 8$  and  $\bar{\alpha}_3(4, 4) = 9$ , as we show in Section 4.3.

**Connection to Ruzsa-Szemerédi Graphs.** A graph is called an  $(r, t)$ -Ruzsa-Szemerédi graph if its edge set can be partitioned into  $t$  edge-disjoint induced matchings, each of size  $r$ . These graphs were introduced in 1978 and have been extensively studied since then. Of particular interest are dense Ruzsa-Szemerédi graphs, with  $r$  and  $t$  large, in terms of  $n$ , the number of vertices. Such graphs have applications in Combinatorics, Complexity theory and Information theory. Also, there are several known interesting constructions, relying on different techniques.

Let  $G$  be a tripartite graph with parts  $R, C, W$  of cardinalities  $n, n, N$  respectively. Let  $T$  be a  $G$ -star-free family of edge disjoint triangles in  $G$ . Let  $F$  be the bipartite graph with parts  $R$  and  $C$  where there is an edge between  $r \in R$  and  $c \in C$  iff there is some  $b \in W$  such that  $(r, c, b) \in T$ . Then  $F$  is the union of at most  $N$  edge-disjoint induced matchings, since all the edges that correspond to a given  $b \in W$  form an induced matching.

This construction can easily be reversed: Let  $F$  be a subgraph of  $K_{n,n}$  that is the union of  $N$  edge disjoint induced matchings, with a total of  $\bar{\alpha}$  edges. We can construct a tripartite  $G$  (a subgraph of  $K_{n,n,N}$ ) that contains a family of  $\bar{\alpha}$  pairwise disjoint triangles, and has no  $G$ -stars. We conclude that

► **Observation 16.** *Let  $n \leq N$  be positive integers, then  $\bar{\alpha}_3(n, N)$  is the largest number of edges in a union of  $N$  edge-disjoint induced matchings in  $K_{n,n}$ .*

This observation exhibits a strong connection between (i) The problem of constructing dense  $(r, t)$ -Ruzsa-Szemerédi graphs, and (ii) The construction of a large star-free subset  $S \subseteq [n] \times [n] \times [t]$  that meets every line at most once. The two problems differ only slightly. In one, the underlying graph is bipartite and in the other all induced matching must have the same cardinality. But these differences can be bridged quite easily, as observed in the following lemma.

► **Lemma 17.** **1.**  *$(r, t)$ -Ruzsa-Szemerédi graphs on  $n$  vertices imply  $\bar{\alpha}_3(\frac{n}{2}, t) \geq \frac{rt}{2}$ .*  
**2.**  *$\bar{\alpha}_3(n, t) \geq rt$  implies that there exists a  $(\frac{r}{2}, t)$ -Ruzsa-Szemerédi graph on  $n$  vertices.*

**Proof.** For the first claim, let  $G = (V, E)$  be a  $(r, t)$ -Ruzsa-Szemerédi graph on  $n$  vertices, and let  $E_1, E_2, \dots, E_t$  be the partition of  $E$  into induced matchings. We can find (e.g., by a random choice) a subset  $A \subset V$  of  $\lfloor \frac{n}{2} \rfloor$  vertices, so that at least  $|E|/2$  edges are in the cut  $C = (A, \bar{A})$ . Also,  $C \cap E_1, C \cap E_2, \dots, C \cap E_t$  is a partition of the edges of the bipartite graph  $(A, \bar{A}, C)$  into  $t$  disjoint induced matchings. Therefore,  $\bar{\alpha}_3(\frac{n}{2}, t) \geq \frac{rt}{2}$ .

For the second part, suppose that  $\bar{\alpha}_3(n, t) \geq rt$ . Namely, there is a collection of disjoint induced matchings  $M_1, \dots, M_t \subseteq E(K_{n,n})$  with  $\sum_1^t |M_i| \geq rt$ . We split each  $M_i$  into  $\lfloor \frac{2|M_i|}{r} \rfloor$  sets of  $\geq r/2$  edges each. Note that  $\sum_1^t a_i \geq rt$  implies that  $\sum_1^t \lfloor \frac{2a_i}{r} \rfloor \geq t$  and a subset of an induced matching is an induced matching, so we finally have a family of at least  $t$  disjoint induced matchings each of size  $\frac{r}{2}$ . ◀

### 3.1.1 Application to Shared Directional Multi-channels

Ruzsa-Szemerédi graphs have various applications in several fields [36, 4, 33, 3, 24, 12]. In [12] they are applied to Information Theory, and the study of *shared directional multi-channels*, a subject that is strongly related to communication complexity. Such a channel is comprised of a set of inputs and a set of outputs, to which are connected transmitters and receivers respectively. Associated with each input is a set of outputs, that receive any signal placed at that input. A message is received successfully at an output of the channel if and only if it is addressed to the receiver connected to that output and no other signals concurrently reach that output. Therefore, when communicating over a shared channel, we want the edges (corresponding to messages sent in one round) to form an induced matching. The challenge is to partition  $K_{n,n}$  into families of pairwise disjoint induced matchings. The number of parts correspond to the number of receivers allowed at each output, and the number of matchings in each partition corresponds to the number of rounds.

The relation to communication complexity is as follows: A  $c$ -bit communication protocol for any injection  $A : [n] \times [n] \rightarrow [N]$  induces a partition of  $K_{n,n}$  into  $c$  such families of disjoint induced matchings. Thus, such a communication protocol, gives an  $N$  round protocol for the shared directional multi-channel, with  $c$  receivers per station, and vice-versa.

In constructing a shared directional multi-channel, we seek to minimize the number of rounds required for a given number of transmitters. Alon, Moitra, and Sudakov [4] showed that for any  $\epsilon > 0$  there is partition of  $K_{n,n}$  into at most  $2^{O(\frac{1}{\epsilon})}$  graphs each of which is a family of at most  $O(n^{1+\epsilon})$  induced matchings. This gives an  $O(n^{1+\epsilon})$  round protocol for shared directional multi-channel with  $2^{O(\frac{1}{\epsilon})}$  receivers.

Translated to the language of NOF protocols and combining with Corollary 34 (see Section 5.3 in the sequel), we conclude:

► **Theorem 18.** *For all  $\epsilon > 0$  and all large enough  $n$ ,  $2^{O(\frac{1}{\epsilon})} \geq \chi_3(n, n^{1+\epsilon}) \geq \Omega(\log \frac{1}{\epsilon})$ .*

### 3.2 A Characterization of $\alpha_k(f_{k,T}^{\mathbb{Z}_2^n})$

In this section we focus on the problem  $f_{k,T}^G$  for the abelian group  $\mathbb{Z}_2^n$ . In other words, we study the permutation  $f_{k,T}^{\mathbb{Z}_2^n}$ . We give an alternative characterization of  $\alpha_3(f_{3,T}^{\mathbb{Z}_2^n})$  which brings forth the relation between this problem and several known combinatorial objects. The complexity of  $f_{k,T}^{\mathbb{Z}_2^n}$  is independent of  $T$ , so we will omit the subscript  $T$  in this section. Also, throughout this section we let  $A_k^G = A(f_k^G)$ .

Let  $X \subset \mathbb{Z}_4^3$ . We call a subset of  $W \subseteq \mathbb{Z}_4^n$  *X-free* if for every three distinct members  $x, y, z \in W$  there is an index  $1 \leq i \leq n$  for which  $(x_i, y_i, z_i) \notin X$ .

► **Theorem 19.** *Let*

$$X = \{(0, 0, 0), (1, 1, 1), (2, 2, 2), (3, 3, 3), (0, 1, 2), (1, 0, 3), (2, 3, 0), (3, 2, 1)\} \subset \mathbb{Z}_4^3,$$

*then  $\alpha_3(A_3^{\mathbb{Z}_2^n})$  is the largest cardinality of an X-free subset of  $\mathbb{Z}_4^n$ .*

**Proof.** Recall that  $\alpha_3(A_3^{\mathbb{Z}_2^n})$  is the largest cardinality of an  $A_n$ -star free subset of  $(\mathbb{Z}_2^n)^2$ , where  $A_n = A_3^{\mathbb{Z}_2^n}$ . So it suffices to find a bijection  $\psi$  from  $(\mathbb{Z}_2^n)^2$  to  $\mathbb{Z}_4^n$  such that  $S \subseteq (\mathbb{Z}_2^n)^2$  is mapped to an X-free set if and only if  $S$  is  $A_n$ -star free.

We define  $\psi$  for  $n = 1$  and extend it entry-wise to a mapping from  $(\mathbb{Z}_2^n)^2$  to  $\mathbb{Z}_4^n$ . The definition for  $n = 1$  is as follows:  $\psi(0, 0) = 0$ ,  $\psi(0, 1) = 1$ ,  $\psi(1, 0) = 2$  and  $\psi(1, 1) = 3$ .

We need to show that if  $(x_1, y_1), (x_2, y_2), (x_3, y_3) \in (\mathbb{Z}_2^n)^2$  is a  $A_n$ -star, then every coordinate in  $(\psi(x_1, y_1), \psi(x_2, y_2), \psi(x_3, y_3))$  belongs to  $X$ , and vice versa. Since the map  $\psi$  is defined coordinate-wise it suffices to check this for  $n = 1$ . A triple  $(x_1, y_1), (x_1 + d, y_1), (x_1 + d', y_1)$  is a (trivial or non-trivial) star in  $A_1$  iff  $x_1 + (y_1 + d) = (x_1 + d') + y_1$ , i.e.,  $d = d'$ , and thus an  $A_1$ -star is a triple of the form  $(x_1, y_1), (x_1, y_1 + d), (x_1 + d, y_1)$ . If  $d = 0$  then obviously  $(\psi(x_1, y_1), \psi(x_1, y_1 + d), \psi(x_1 + d, y_1)) \in \{(0, 0, 0), (1, 1, 1), (2, 2, 2), (3, 3, 3)\} \subset X$ . When  $d = 1$  there are four cases to check:

1.  $x_1 = 0$  and  $y_1 = 0$  then  $(\psi(x_1, y_1), \psi(x_1, y_1 + d), \psi(x_1 + d, y_1)) = (0, 1, 2) \in X$ .
2.  $x_1 = 0$  and  $y_1 = 1$  then  $(\psi(x_1, y_1), \psi(x_1, y_1 + d), \psi(x_1 + d, y_1)) = (1, 0, 3) \in X$ .
3.  $x_1 = 1$  and  $y_1 = 0$  then  $(\psi(x_1, y_1), \psi(x_1, y_1 + d), \psi(x_1 + d, y_1)) = (2, 3, 0) \in X$ .
4.  $x_1 = 1$  and  $y_1 = 1$  then  $(\psi(x_1, y_1), \psi(x_1, y_1 + d), \psi(x_1 + d, y_1)) = (3, 2, 1) \in X$ .

On the other hand it is not hard to check that for each  $(a, b, c) \in X$  the triplet  $\psi^{-1}(a), \psi^{-1}(b), \psi^{-1}(c)$  is a star in  $A_1$  or  $a = b = c$ . This proves the claim. ◀

Fix an integer  $s \geq 2$  and let  $HJ(n, s)$  denote the largest size of a  $Y_s$ -free subset of  $[s]^n$ , where  $Y_s$  is the following set of  $s$ -tuples:  $\{(1, \dots, s)\} \cup \{(i, i, \dots, i) \mid i = 1, 2, \dots, s\}$ . The density Hales-Jewett theorem states that  $HJ(n, s) = o(s^n)$  for every fixed  $s$  [19, 31]. Theorem 19, and the observation that the first three coordinates of the 4-tuples in  $Y_4$  all belong to  $X$ , imply that  $\alpha_3(A_3^{\mathbb{Z}_2^n}) \leq HJ(n, 4)$ .

The cap-set problem for  $\mathbb{Z}_4^n$  also belongs to the same circle of problems. It concerns the largest size of an arithmetic-triple-free set in  $\mathbb{Z}_4^n$ . We mention in passing the recent breakthrough [14, 17] in this area which showed that this size is at most  $4^{(\gamma+o(1))n}$  with  $\gamma \approx 0.926$ . Let  $Z \subset \mathbb{Z}_4^3$  be the set of all ordered triplets  $(a, b, c) \in \mathbb{Z}_4^3$  satisfying  $a + c = 2b$ . The cap set problems concerns exactly the largest possible cardinality of a  $Z$ -free subset of  $\mathbb{Z}_4^n$ . Since  $X \subset Z$  it follows that this size is bounded by  $\alpha_3(A_3^{\mathbb{Z}_2^n})$ .

The proof of Theorem 19 extends verbatim to general  $k \geq 3$ . It yields a subset  $X \subset \mathbb{Z}_{2^{k-1}}^k$  such that  $\alpha_k(A_k^{\mathbb{Z}_k^n})$  is the largest cardinality of an  $X$ -free subset of  $\mathbb{Z}_{2^{k-1}}^n$ .

By taking  $X$  that includes all vectors  $(a, a, \dots, a) \in \mathbb{Z}_{2^{k-1}}^k$  for  $a \in \mathbb{Z}_{2^{k-1}}$  and the vector  $(0, 1, 2, 4, \dots, 2^{k-2})$  we can maintain the relation between  $\alpha_k(A_k^{\mathbb{Z}_k^n})$  and the density Hales-Jewett theorem for every  $k$ .

## 4 Upper Bounds

### 4.1 An Algorithmic Protocol for Exact- $T$ over $\mathbb{Z}^d$

The aim of this section is to give the first algorithmic protocol for Exactly- $n$  as well as other Exact- $T$  functions. Our protocol is explicit, and does not rely on a construction of a large set without a 3-term AP. We only appeal to the elementary fact that no sphere can contain three equally spaced colinear points. The algorithm has two main steps. We first provide a very efficient protocol for Exact- $T$  over  $\mathbb{Z}^d$ , whose cost grows only logarithmically with  $d$ .

Let  $f : ([m]^d)^3 \rightarrow \{0, 1\}$  be defined via  $f(x, y, z) = 1$  if and only if  $x + y + z = T$ , where  $T \in \mathbb{Z}^d$  is some fixed vector. We provide an explicit NOF protocol for  $f$  whose cost is only  $O(\log md)$ . In words, players try to compute the vector  $x + 2y + 3z$  “to the best of their knowledge” and then they compare notes.

1. Player 1 computes  $v_x = T - y - z + 2y + 3z$ .
2. Player 2 computes  $v_y = x + 2(T - x - z) + 3z$ .
3. Player 3 computes  $v_z = x + 2y + 3(T - x - y)$ .
4. Player 1 writes  $\|v_x\|_2^2$  on the blackboard.
5. Player 2 writes 1 or 0 on the blackboard depending on whether  $\|v_y\|_2^2 = \|v_x\|_2^2$ .
6. Player 3 writes 1 or 0 on the blackboard depending on whether  $\|v_z\|_2^2 = \|v_x\|_2^2$ .
7. The protocol outputs 1 if the last two bits were both equal to 1, and 0 otherwise.

The cost of the above protocol is essentially determined by the largest possible value of  $\|v_x\|_2^2$  in step 4 which is at most  $O(m^2d)$ . Therefore, this cost does not exceed  $O(\log md)$ . We turn to prove correctness.

► **Lemma 20.** *The above protocol is correct.*

**Proof.** First note that the protocol outputs 1 if and only if  $\|v_x\|_2^2 = \|v_y\|_2^2 = \|v_z\|_2^2$ . Also,  $v_x + v_z = 2v_y$ , so that this condition holds only if all three vectors are equal, in which case  $T - x - y - z = 0$ . ◀

► **Remark (More general protocols).** Several variations on the above theme suggest themselves. Fix integers  $a, b, c \in \mathbb{Z}$  and a  $d \times d$  positive definite matrix  $D$  with integer entries. The players compute  $a(T - y - z) + by + cz$ ,  $ax + b(T - x - z) + cz$  and  $ax + by + c(T - x - y)$ , and rather than comparing the values of  $\|v\|_2$ , they consider the values of  $vDv^t$ . We wonder if these, or similar variations can together improve the complexity of the protocol.

### 4.2 Algorithmic Protocols for Exactly- $N$ and $f_{k,T}^G$ over $\mathbb{Z}_m^n$

We seek algorithmically explicit protocols for Exactly- $N$ <sup>2</sup>, namely for the function  $f : [N]^3 \rightarrow \{0, 1\}$  such that  $f(x, y, z) = 1$  if and only if  $x + y + z = N$ , i.e., the exact-T problem over  $\mathbb{Z}$

<sup>2</sup> Since  $n$  is used as an exponent in this section we use  $N$  for the size of input in Exactly- $N$ .



or equivalently over  $\mathbb{Z}_N$ . We can give an efficient protocol to this problem by reduction to the protocol in the previous section, even though when applied directly to  $\mathbb{Z}_N$  they give no improvement over the trivial protocol.

First fix a base  $m$  and let  $n = 1 + \lceil \log_m N \rceil$ . Consider the base- $m$  representation on the elements of  $[N]$ . Given a representation  $x \in \{0, 1, \dots, m-1\}^n$  of a number base  $m$ , for convenience we consider  $x_1$  as the least significant digit. Note that all representations are of length  $n$ , if a number is small its representation is padded with zeros. The following protocol solves the Exactly- $N$  problem in these settings. Let  $T$  be the base- $m$  representation of  $N$ .

1. Player 1 computes the vector  $C \in \{0, 1, 2\}^m$  defined as follows: the  $i$ -th entry of  $C$  is equal to  $k \in \{0, 1, 2\}$  satisfying

$$T_i + (k-1)m < y_i + z_i + C_{i-1} \leq T_i + km,$$

where addition is over  $\mathbb{Z}$ , and we define  $C_0 = 0$ .

2. Denote by  $C_x$  the carry vector computed by Player 1 in step 1. Player 2 and 3 compute corresponding vectors  $C_y$  and  $C_z$ , in a similar way.
3. Player 1 writes  $C = C_x$  on the board.
4. Player 2 and 3, in turn, write 1 on the board if and only if their vector  $C_y$  ( $C_z$ ) is equal to  $C_x$ .
5. If the last two bits written on the board are equal to 1, continue. Otherwise output 0 and terminate.
6. All players compute (in private) the vector  $T'_i = T_i + mC_i - C_{i-1}$ , for  $i = 1, \dots, n$ .
7. The players run a protocol for the exact-T problem over  $\mathbb{Z}^n$  with  $x, y, z$  and  $T'$ .

The cost of the above protocol is  $O(n+2)$  for steps 1-5, plus the cost of the protocol used in step 7. The cost is thus  $O(n + \log mn)$  if the players use the protocol from Section 4.1 in the last step. We prove next that this protocol is correct.

► **Lemma 21.** *The above protocol is correct.*

**Proof.** First assume  $N = x + y + z$  over  $\mathbb{Z}$ . It is easy to verify the correctness of the protocol in this case, except maybe step 5. The correctness of step 5 follows from the following simple observation: assume  $x_i + y_i + z_i + C_{i-1} = T_i + km$  (over  $\mathbb{Z}$ ) for  $k \in \{0, 1, 2\}$ , then it must be that the sum of any pair of  $x_i, y_i, z_i$  and  $C_{i-1}$  is larger than  $T_i + (k-1)m$  and at most  $T_i + km$ . Now consider the case  $T \neq x + y + z$ . If the protocol rejects on step 5 then obviously this is correct. If it does not reject then all players compute the same vector  $T'$ , and  $x + y + z = N$  over  $\mathbb{Z}$  if and only if  $x + y + z = T'$  over  $\mathbb{Z}^n$ . The correctness now follows from the correctness of the protocol over  $\mathbb{Z}^n$ . ◀

The above protocol for Exactly- $N$  is correct for any choice of base  $m$ . To get an efficient protocol we optimize the choice of  $m$ . The running time of the protocol is  $O(n + \log mn) = O(n + \log m)$ . Since  $m^n = N$ , we get that  $\log N = n \log m$ , and thus the optimal choice is roughly  $m = 2^{\sqrt{\log N}}$  which gives a running time of  $O(\sqrt{\log N})$ .

► **Remark (The group  $\mathbb{Z}_m^n$ ).** The above protocol can also be adapted for  $\mathbb{Z}_m^n$  (with addition modulo  $m$ ). The idea is very similar, the only difference is that in the first steps Player 1 computes the vector  $I_x \in \{0, 1, 2\}^n$  defined as follows: the  $i$ -th entry of  $I_x$  is equal to  $k \in \{0, 1, 2\}$  satisfying

$$T_i + (k-1)m < y_i + z_i \leq T_i + km,$$

where addition is over  $\mathbb{Z}$ . The other two players compute analogous vectors.



### 4.3 A Protocol for $f_{k,T}^G$ over $\mathbb{Z}_2^n$

In this section we focus on the exact- $T$  problem for the abelian group  $\mathbb{Z}_2^n$ . In other words, we study the permutation  $f_{k,T}^{\mathbb{Z}_2^n}$ . First we prove a lower bound on  $\alpha_3(f_{3,T}^{\mathbb{Z}_2^n})$ , and then show that this lower bound implies the existence of an efficient protocol for  $f_{3,T}^{\mathbb{Z}_2^n}$ . The complexity of  $f_{k,T}^{\mathbb{Z}_2^n}$  is independent of  $T$ , so we can and will omit the subscript  $T$  in this section. Throughout this subsection we let  $A_k^G = A(f_k^G)$ .

First we prove that  $A_k^{\mathbb{Z}_2^n}$ -star freeness is preserved under tensor product. Let  $S \subset (\mathbb{Z}_2^n)^{k-1}$ , denote by  $S \otimes S$  the subset of  $(\mathbb{Z}_2^{2n})^{k-1}$  comprised of all vectors  $(x_1, y_1, \dots, x_{k-1}, y_{k-1})$  such that  $x_i, y_i \in S$  for  $i = 1, \dots, k-1$ .

► **Lemma 22.** *If  $S$  is  $A_k^{\mathbb{Z}_2^n}$ -star free then  $S \otimes S$  is  $A_k^{\mathbb{Z}_2^{2n}}$ -star free.*

**Proof.** Let  $A = A_k^{\mathbb{Z}_2^{2n}}$  and let

$$(z_1, \dots, z_{k-1}), (z_1 + d, \dots, z_{k-1}), \dots, (z_1, \dots, z_{k-1} + d)$$

be an  $A$ -star in  $S \times S$ , where for each  $1 \leq i \leq k-1$ ,  $z_i = (x_i, y_i)$  with  $x_i, y_i \in S$ . Denote also  $d = (d^1, d^2)$  where  $d^1, d^2 \in \mathbb{Z}_2^n$ . Then either

$$(x_1, \dots, x_{k-1}), (x_1 + d^1, \dots, x_{k-1}), \dots, (x_1, \dots, x_{k-1} + d^1)$$

is an  $A_k^{\mathbb{Z}_2^n}$ -star in  $S$ , or

$$(y_1, \dots, y_{k-1}), (y_1 + d^2, \dots, y_{k-1}), \dots, (y_1, \dots, y_{k-1} + d^2)$$

is an  $A_k^{\mathbb{Z}_2^n}$ -star in  $S$ , since either  $d^1 \neq 0$  or  $d^2 \neq 0$ . ◀

It follows that if, for some fixed  $m$ , we can find a large  $A_k^{\mathbb{Z}_2^m}$ -star free subset  $S$ , then tensor powers of  $S$  are large  $A_k^{\mathbb{Z}_2^n}$ -star free sets. We show:

► **Lemma 23.**  $\alpha_3(A_3^{\mathbb{Z}_3^2}) = \alpha_3(n, n) = 8$ .

Together with Lemma 22 this yields:

► **Corollary 24.** *For every integer  $n \geq 2$ , there holds  $\alpha_3(A_3^{\mathbb{Z}_3^{2n}}) \geq 2^{3n/2}$ .*

**Proof.** Let  $S$  be a star-free subset in  $A_3^{\mathbb{Z}_3^2}$  of cardinality  $8 = 4^{3/2}$  as in Lemma 23. The claim follows by taking the tensor powers of  $S$  as in Lemma 22. ◀

**Proof of Lemma 23.** We denote the elements of  $\mathbb{Z}_3^2$  as follows  $(0, 0) = 0$ ,  $(0, 1) = 1$ ,  $(1, 0) = 2$  and  $(1, 1) = 3$ . The matrix associated with  $A_3^{\mathbb{Z}_3^2}$  is:

$$\begin{matrix} \mathbf{0} & \mathbf{1} & \mathbf{2} & \mathbf{3} \\ 1 & 0 & \mathbf{3} & \mathbf{2} \\ \mathbf{2} & \mathbf{3} & 0 & 1 \\ 3 & 2 & \mathbf{1} & \mathbf{0} \end{matrix}$$

The 8 entries in bold form a star-free set, so that  $\alpha_3(A_3^{\mathbb{Z}_3^2}) \geq 8$ , and consequently  $\alpha_3(4, 4) \geq 8$ . One can verify that in fact  $\alpha_3(4, 4) = \alpha_3(A_3^{\mathbb{Z}_3^2}) = 8$ . To see this first notice that if there is a star-free subset of cardinality 9 then one of the values must appear three times which already determines 10 out of the 16 entries. One can now rule out the existence of a size 9 star-free subset by exhaustive search. ◀

It is interesting to determine  $\alpha_k(n, n)$  for some small values of  $n$ . For example:

- Determine  $\alpha_3(8, 8)$ , in particular compute  $\alpha_3(A_3^{\mathbb{Z}_3^8})$ .
- Determine  $\alpha_k(4, 4)$ , in particular compute  $\alpha_k(A_k^{\mathbb{Z}_k^4})$ , for  $k > 3$ .

It is interesting to note that, while as shown,  $\alpha_3(4, 4) = 8$ , there holds  $\bar{\alpha}_3(4, 4) = 9$ . The fact that  $\bar{\alpha}_3(4, 4) \leq 9$  is easy to verify, and the following example shows the equality:

1	*	*	3
*	1	*	4
*	*	1	2
2	3	4	*

Thus, continuing the discussion at the end of Section 3.1,  $\bar{\alpha}_3(n, N)$  and  $\alpha_3(n, N)$  need not be equal when  $N < 2n - 1$ .

The following theorem shows that for groups,  $\alpha_k$  (the size of the largest 1-monochromatic cylinder intersection) completely characterizes  $\chi_k$  (the minimum number of cylinder intersections that partition the 1's). The proof is a simple generalization of Theorem 4.3 in [13].

► **Theorem 25.** *If  $G$  is a group of order  $n$ , then*

$$\chi_k(f_k^G) \leq O\left(\frac{kn^{k-1} \log n}{\alpha_k(f_k^G)}\right).$$

**Proof.** The proof is in two steps:

**Step I:**  $A$ -star freeness is preserved under translation, where  $A = A_k^G$ . Indeed, let  $S \subset G^{k-1}$  and let  $\mathbf{a} = (a_1, \dots, a_{k-1}) \in G^{k-1}$ . If

$$(x_1, \dots, x_{k-1}), (x_1 + d, \dots, x_{k-1}), \dots, (x_1, \dots, x_{k-1} + d)$$

is an  $A$ -star in  $S + \mathbf{a}$ , then

$$(x_1, \dots, x_{k-1}) - \mathbf{a}, (x_1 + d, \dots, x_{k-1}) - \mathbf{a}, \dots, (x_1, \dots, x_{k-1} + d) - \mathbf{a}$$

is an  $A$ -star in  $S$ .

**Step II:** Every  $S \subset G^{k-1}$  has  $O(\frac{kn^{k-1} \log n}{|S|})$  translates whose union covers all of  $G^{k-1}$ . This follows from the integrality gap for covering [28], but for completeness here is a proof. Pick at random  $t$  translates  $\mathbf{a}_1, \dots, \mathbf{a}_t \in [n]^{k-1}$  of  $S$ . The probability that a given element  $\mathbf{x} \in [n]^{k-1}$  is covered by a random translate of  $S$  is exactly  $\frac{|S|}{n^{k-1}}$ . Therefore, and since the translates are picked independently uniformly at random, the expected number of uncovered elements of  $G^{k-1}$  is

$$n^{k-1} \cdot \left(1 - \frac{|S|}{n^{k-1}}\right)^t.$$

Taking  $t = O(\frac{kn^{k-1} \log n}{|S|})$  makes the expectation less than 1, which proves the lemma. ◀

► **Corollary 26.**  $\chi_3(f_3^{\mathbb{Z}_3^m}) \leq O(m \cdot 2^{m/2})$ .

The bound in Corollary 26 is similar to the bound of Ada, Chattopadhyay, Fawzi and Nguyen [1] for the case  $k = 3$ , with slight improvement in the log factors. Ada et al. proved  $\chi_3(f_3^{\mathbb{Z}_3^m}) \leq O(m^{k+1} 2^{m/2^{k-2}})$ , by observing that this function is a composed function of the form  $NOR \circ XOR$  and giving non trivial protocols for such cases.

Note that the proof of Theorem 25 yields a cover of  $[n]^{k-1}$  by  $A$ -star free sets, but this is easily turned into a partition, since a subset of an  $A$ -star free set is also  $A$ -star free. Therefore, any lower bound on  $\alpha_k(f_k^G)$  can be translated into an upper bound on  $\chi_k(f_k^G)$  which in turn implies an efficient (non-explicit) protocol for  $f_k^G$  (By Theorem 5). Another interesting consequence of Theorem 25 is that any lower bound on  $\chi_k(f_k^G)$  significantly larger than  $\log n$  improves the known bounds for the size of a corner-free subset of  $G$ . This clearly boosts our interest in the multiparty communication complexity of  $f_k^G$ .

We wonder whether there are analogs of Theorem 25 for every permutation.

► **Question 27.** *How large can  $\chi_k(A) \cdot \alpha_k(A)/n^{k-1}$  be for an arbitrary permutation  $A$ ?*

## 5 Lower Bounds

### 5.1 Nonconstructive Lower Bounds

We first prove a nearly tight but nonconstructive lower bound on the communication complexity of random high-dimensional permutations.

► **Theorem 28.** *For every integer  $k \geq 3$ , and for most  $(k - 1)$ -dimensional permutations  $f : [n]^k \rightarrow \{0, 1\}$ ,*

$$\log \chi_k(f) \geq \Omega\left(\frac{\log n}{k}\right).$$

**Proof.** The lower bound on the number of high-dimensional permutations was recently improved by Keevash [25] who showed that there are at least  $2^{\Omega(n^d \log n)}$   $d$ -dimensional permutations. If we view a permutation as a map  $[n]^k \rightarrow \{0, 1\}$ , this means at least  $2^{\Omega(n^{k-1} \log n)}$  permutations. In the spirit of the proof of Lemma 3.5 in [7], we now estimate the number of such permutation for which  $\chi_k(f)$  is bounded. Note that we cannot simply use the estimate from [7] since it only works for functions  $f : [n]^{k-1} \times [N] \rightarrow \{0, 1\}$  with  $N$  that is much smaller than  $n$ , roughly  $N \leq \sqrt{\frac{n}{k}}$ .

Let  $f : [n]^k \rightarrow \{0, 1\}$  be a  $(k - 1)$ -dimensional permutation, and let  $\{C_1, \dots, C_\chi\}$  be a partition of  $f^{-1}(1)$  into  $\chi = \chi_k(f)$  cylinder intersections. For  $i \in [k]$  define a function  $A_i : [n]^{k-1} \rightarrow [\chi]$  as follows: For  $a = (a_1, \dots, a_{k-1}) \in [n]^{k-1}$ , let  $L = (a, i)$  be a line in  $[n]^{k-1}$ . There is a unique 1 entry in  $L$  and this entry is in exactly one of the cylinder intersections  $\{C_1, \dots, C_\chi\}$ , say  $C_j$ . In this case we define  $A_i(a_1, \dots, a_{k-1}) = j$ .

As seen in the proof of Theorem 5, it is possible to recover  $f$  from knowledge of the functions  $A_1, \dots, A_k$ . Namely,  $f(x_1, \dots, x_k) = 1$  if and only if all the values  $A_i(x_1, \dots, x_{i-1}, x_{i+1}, \dots, x_{k-1})$  for  $i = 1, \dots, k$  are equal. But for every  $i \in [k]$  there are  $\chi^{n^{k-1}}$  possible functions  $A_i : [n]^{k-1} \rightarrow [\chi]$ . Thus, the number of  $(k - 1)$ -dimensional permutations  $f : [n]^k \rightarrow \{0, 1\}$  with  $\chi_k(f) \leq \chi$  is at most  $(\chi^{n^{k-1}})^k = 2^{kn^{k-1} \cdot \log \chi}$ . Combining this with Keevash's lower bound achieves our result. ◀

A simple corollary of Theorem 28, and Theorem 5 is:

► **Corollary 29.** *For every integer  $k \geq 2$ , almost all  $(k - 1)$ -dimensional permutations  $f : [n]^k \rightarrow \{0, 1\}$  satisfy  $D_k(f) \geq \Omega\left(\frac{\log n}{k}\right)$ .*

Theorem 28 proves the lower bound  $\chi_k(f) \geq 2^{\Omega\left(\frac{\log n}{k}\right)}$  for a random permutation  $f : [n]^k \rightarrow \{0, 1\}$ . It is interesting to find out how this extends for a random linjection  $f : [n]^{k-1} \times [N] \rightarrow \{0, 1\}$  with  $n < N$ . It is also interesting to see whether the dependency on  $k$  can be removed.

Finally we turn to the case  $k = 3$ . The number of 2-dimensional permutations (aka Latin squares) is known to be  $((1 + o(1)) \frac{n}{\epsilon^2})^{n^2}$  (see [39]). It follows that for most 2-dimensional permutations  $f$  there holds  $\log \chi_3(f) \geq \frac{1}{3} \log n - \Theta(1)$ .

## 5.2 Lower Bounds for $\chi_k(n, N)$

We prove an upper bound on  $\alpha_k(n, N)$ , using its graph theoretic interpretation from Section 3.1, which implies the corresponding lower bound on  $\chi_k$ . We start with  $k = 3$ :

► **Theorem 30.** *Let  $A : [n] \times [n] \rightarrow [N]$  be a linjection, where  $N \leq n \cdot 2^{c \log^*(n)}$ . Then there exists  $c > 0$  such that  $\alpha_3(A) \leq O\left(\frac{n^2}{2^{c \log^*(n)}}\right)$ .*

The proof of Theorem 30 is an adaptation of Solymosi's [36] simplification of Ajtai and Szemerédi's [2] Corners Theorem. We will use the improved version of the triangle removal lemma [33] due to Fox [18]:

► **Lemma 31** (Triangle removal lemma). *For every  $\epsilon > 0$  there is a  $\delta > 0$  such that every  $n$ -vertex graph with at most  $\delta n^3$  triangles can be made triangle-free by removing  $\epsilon n^2$  edges. Specifically  $\delta^{-1}$  can be taken as a tower of twos of height  $405 \log \epsilon^{-1}$ .*

**Proof of Theorem 30.** Let  $G = G(A)$ ,  $V = V(G)$ . Notice that  $|V| = 2n + N$ . Let  $S \subset [n]^2$  be an  $A$ -star free subset of size  $\alpha_3(A)$ . As in the proof of Theorem 14 we let  $T = \{ \langle x, y, A(x, y) \rangle \mid (x, y) \in S \}$  be the family of triangles in  $G$  that corresponds to  $S$ . Let  $F$  be that subgraph of  $G$  whose edge set is the union of all triangles in  $T$ . This graph contains the  $|S|$  edge-disjoint triangles in  $T$ , and no additional triangles.

Thus, if we denote  $\delta = |S|/|V|^3$  and  $\epsilon = |S|/|V|^2$ , then  $F$  contains exactly  $\delta|V|^3$  triangles and it cannot be made triangle free by removing fewer than  $\epsilon|V|^2$  edges. Lemma 31 yields  $\log^*(\delta^{-1}) \leq 405 \log(\epsilon^{-1})$ , and since  $\delta < \frac{n^2}{(2n+N)^3} < \frac{1}{N}$  we conclude that

$$\epsilon \leq 2^{\frac{1}{405} \log^*(N)}.$$

But  $|S| = \epsilon|V|^2 \leq 9\epsilon N^2$ , so that for  $N \leq 2^{c \log^*(n)} n$ , with  $c = (3 \cdot 405)^{-1}$ ,  $|S| \leq O\left(\frac{n^2}{2^{c \log^*(n)}}\right)$ . ◀

We now state the case of general  $k$ , proved in the full version.

► **Theorem 32.** *For every natural numbers  $k \geq 3$ ,  $n$  and  $N$  it holds that*

$$\alpha_k(n, N) \leq O\left(\frac{kn^{k-2}N}{\log^*(n)}\right).$$

## 5.3 A Lower Bound on $\chi_3(n, N)$

In this section we state our better lower bound for the case  $k = 3$ . The proofs appears in the full version of our paper.

► **Lemma 33.** *Let  $L = \chi_3(n, N)$  for some integers  $N \geq n$ , then  $\log n < (2^{L+1} - 1) \cdot \log(4NL/n)$ . In particular for  $k = 3$ , we have  $\chi_3(n, n) \geq \log \log n - O(\log \log \log n)$ .*

Another simple corollary of Lemma 33 is due to Meshulam and is reproduced in [4].

► **Corollary 34.** *If  $\chi_3(n, N) \leq L$  for some integers  $N \geq n$ , then  $N \geq \frac{1}{4L} \cdot n^{1+1/(2^L-1)}$ .*

**A note on the case  $k > 3$ .** As we have just seen  $\chi_3(A) \geq \Omega(\log \log n)$  for every 2-dimensional permutation  $A$ . It is conceivable that a similar bound holds for higher dimensions as well. This was previously conjectured in [10] for the Exact- $T$  problem. If we try to adapt the proof of Lemma 33 to higher  $k$ , exactly one difficulty arises which we formulate as a question.

► **Question 35.** Let  $S \subseteq [n]^k$  be a set of cardinality  $m$  that meets every line at most once. Determine, or estimate  $\phi_k(n, m)$ , the least possible cardinality  $|\bar{S}|$  of its closure. We use the shorthand  $\phi_k(m)$  when appropriate.

For  $k = 2$  the answer is easy:  $\phi_2(m) = m^2$ , since  $|\bar{S}| = |S|^2$ . But for  $k > 2$  the problem becomes very hard and no lower bound is known. In fact, for  $k \geq 3$ , and for large enough  $m$  there holds  $\phi_k(m) = m$ . In other words, unlike the case  $k = 2$  it may happen that  $\bar{S} = S$  for large  $S$ . For example, as shown in [13],  $\phi_3(m) = m$  when  $m = n^2/2^{\Omega(\sqrt{\log n})}$ , whereas it is shown in [34] that  $\phi_3(m) > m$  when  $m \geq n^2/(\log \log n)^{\frac{1}{22}}$ . For  $k > 3$  the situation is even worse, and all we have are the very weak lower bounds from Section 5.2. Namely, it follows from Theorem 32 that  $\phi_k(m)$  must be larger than  $m$  when  $m \geq \Omega\left(\frac{kn^{k-1}}{\log^r(n)}\right)$ . Proving any non-trivial bounds on  $\phi_k(m)$  is a very interesting challenge. We raise the following conjecture in an attempt to improve the lower bounds on  $\chi_3(n, n)$ :

► **Conjecture 36.** There are constants  $c_1, c_2 > 0$  such that if  $S \subseteq [n]^3$  meets every line at most once, and if  $|S| \geq n^2/(\log \log n)^{c_1}$ , then  $|\bar{S}| \geq n^3/(\log \log n)^{c_2}$ .

## 6 Conclusion and Open Problems

This paper raises numerous open problems. Below we collect some of the major ones and explain some implications that would follow from progress on these questions.

► **Question 37.** Improve the lower bound  $\chi_3(n, n) \geq \Omega(\log \log n)$ .

Any lower bound  $\chi_3(n, n) \geq \omega(\log \log n)$  yields an improvement to the best known bound on the number of colors required to color the  $n \times n$  grid with no monochromatic equilateral right triangles. This subject goes back to Ajtai and Szemerédi's corners theorem [2] and its implications in additive combinatorics due to Solymosi [36]. A lower bound  $\chi_3(n, n) \geq \omega(\log n)$  would improve the best known gap between randomized and deterministic communication complexity in the 3-players NOF model. A lower bound  $\chi_3(n, n) \geq \Omega(\log n \cdot \log \log n)$  will improve the best known upper bound on the size of corner-free subsets of  $G^2$  for any abelian group  $G$ . And finally, a lower bound  $\chi_3(n, n) \geq \Omega(\log^2 n)$  will improve the best bounds on the size of a subset of  $\mathbb{Z}_n$  with no three-term arithmetic progression. This is a classic problem that goes back at least to the 1950's [32].

► **Question 38.** Improve the upper bound  $\chi_3(n, n) \leq 2^{O(\sqrt{\log n})}$ .

The construction of denser Ruzsa-Szemerédi graphs than currently known. Namely,  $n$ -vertex graphs which are the disjoint union of  $n$  induced matchings, all of the same size  $r$ . This, in turn, reflects on the many applications of these. This would also improve our understanding regarding the limits of the triangle removal lemma; note that the current gaps between the bound in this lemma are huge.

► **Question 39.** Improve the bounds on  $\chi_k(n, n)$  for  $k > 3$ .

► **Question 40.** Improve the bounds on  $\alpha_k(n, n)$  for  $k > 3$ .

That would improve our state of knowledge regarding the bounds for the hypergraph removal lemma. It is also interesting to determine  $\alpha_k(n, n)$  for some small values of  $n$ . For example: Determine  $\alpha_3(8, 8)$ , and in particular compute  $\alpha_3(A_3^{\mathbb{Z}_3^2})$ , or Determine  $\alpha_k(4, 4)$ , and in particular compute  $\alpha_k(A_k^{\mathbb{Z}_2^2})$  for  $k > 3$ .

► **Question 41.** *What is the relationship between  $\bar{\alpha}_k(n, N)$  and  $\alpha_k(n, N)$  in the whole range  $n \leq N \leq (k - 1)(n - 1)$ ?*

---

#### References

- 1 A. Ada, A. Chattopadhyay, O. Fawzi, and P. Nguyen. The NOF multiparty communication complexity of composed functions. *computational complexity*, 24(3):645–694, 2015.
- 2 M. Ajtai and E. Szemerédi. Sets of lattice points that form no squares. *Stud. Sci. Math. Hungar*, 9(1975):9–11, 1974.
- 3 N. Alon. Testing subgraphs in large graphs. *Random Structures & Algorithms*, 21(3-4):359–370, 2002.
- 4 N. Alon, A. Moitra, and B. Sudakov. Nearly complete graphs decomposable into large induced matchings and their applications. In *Proceedings of the forty-fourth annual ACM symposium on Theory of computing*, pages 1079–1090. ACM, 2012.
- 5 L. Babai, T. Hayes, and P. Kimmel. The cost of the missing bit: communication complexity with help. *Combinatorica*, 21:455–488, 2001.
- 6 L. Babai, N. Nisan, and M. Szegedy. Multiparty Protocols, Pseudorandom Generators for Logspace, and Time-Space Trade-offs. *Journal of Computer and System Sciences*, 45:204–232, 1992.
- 7 P. Beame, M. David, T. Pitassi, and P. Woelfel. Separating deterministic from randomized NOF multiparty communication complexity. In *Proceedings of the 34th International Colloquium On Automata, Languages and Programming*, Lecture Notes in Computer Science. Springer-Verlag, 2007.
- 8 P. Beame, T. Pitassi, and N. Segerlind. Lower bounds for Lovász-Schrijver systems and beyond follow from multiparty communication complexity. *SIAM Journal on Computing*, 37(3):845–869, 2006.
- 9 F. A. Behrend. On sets of integers which contain no three terms in arithmetical progression. *Proceedings of the National Academy of Sciences*, 32(12):331–332, 1946.
- 10 R. Beigel, W. Gasarch, and J. Glenn. The multiparty communication complexity of Exact-T: Improved bounds and new problems. In *International Symposium on Mathematical Foundations of Computer Science*, pages 146–156. Springer, 2006.
- 11 K. Bibak. Additive Combinatorics: With a View Towards Computer Science and Cryptography - An Exposition. In *Number Theory and Related Fields, In Memory of Alf van der Poorten*, pages 99–128. Springer, 2013.
- 12 Y. Birk, N. Linial, and R. Meshulam. On the uniform-traffic capacity of single-hop interconnections employing shared directional multichannels. *IEEE Transactions on Information Theory*, 39(1):186–191, 1993.
- 13 A. Chandra, M. Furst, and R. Lipton. Multi-party protocols. In *Proceedings of the 15th ACM Symposium on the Theory of Computing*, pages 94–99. ACM, 1983.
- 14 E. Croot, V. Lev, and P. Pach. Progression-free sets in  $\mathbb{Z}_4^n$  are exponentially small. *arXiv preprint*, 2016. [arXiv:1605.01506](https://arxiv.org/abs/1605.01506).
- 15 Z. Dvir. On the size of Kakeya sets in Finite Fields. *J. Amer. Math Soc.*, pages 1093–1097, 2009.
- 16 M. Elkin. An improved construction of progression-free sets. In *Proceedings of the twenty-first annual ACM-SIAM symposium on Discrete Algorithms*, pages 886–905. Society for Industrial and Applied Mathematics, 2010.

- 17 J. S. Ellenberg and D. G. On large subsets of  $\mathbb{F}_q^n$  with no three-term arithmetic progression. *Annals of Mathematics*, 185(1):339–343, 2017.
- 18 J. Fox. A new proof of the graph removal lemma. *Annals of Mathematics*, pages 561–579, 2011.
- 19 H. Furstenberg and Y. Katznelson. A density version of the Hales-Jewett theorem. *Journal d'Analyse Mathématique*, 57(1):64–119, 1991.
- 20 R. Graham and J. Solymosi. Monochromatic equilateral right triangles on the integer grid. In *Topics in discrete mathematics*, pages 129–132. Springer, 2006.
- 21 R. L. Graham, B. L. Rothschild, and J. H. Spencer. *Ramsey theory*, volume 20. John Wiley & Sons, 1990.
- 22 V. Grolmusz. The BNS lower bound for multi-party protocols is nearly optimal. *Information and computation*, 112(1):51–54, 1994.
- 23 J. Håstad and M. Goldmann. On the power of small-depth threshold circuits. *Computational Complexity*, 1:113–129, 1991.
- 24 J. Håstad and A. Wigderson. Simple analysis of graph tests for linearity and PCP. *Random Structures & Algorithms*, 22(2):139–160, 2003.
- 25 P. Keevash. The existence of designs II. *arXiv preprint*, 2018. [arXiv:1802.05900](https://arxiv.org/abs/1802.05900).
- 26 E. Kushilevitz and N. Nisan. *Communication Complexity*. Cambridge University Press, 1997.
- 27 N. Linial and Z. Luria. An upper bound on the number of high-dimensional permutations. *Combinatorica*, 34(4):471–486, 2014.
- 28 L. Lovász. On the ratio of optimal integral and fractional covers. *Discrete Mathematics*, 13:383–390, 1975.
- 29 S. Lovett. Additive Combinatorics and its Applications in Theoretical Computer Science. *Theory of Computing, Graduate Surveys*, 8:1–55, 2017.
- 30 N. Nisan and A. Wigderson. Rounds in communication complexity revisited. In *Proceedings of ACM STOC*, pages 419–429. ACM, 1991.
- 31 D.H.J. Polymath. A new proof of the density Hales-Jewett theorem. *arXiv preprint*, 2009. [arXiv:0910.3926](https://arxiv.org/abs/0910.3926).
- 32 K. F. Roth. On certain sets of integers. *Journal of the London Mathematical Society*, 1(1):104–109, 1953.
- 33 I. Ruzsa and E. Szemerédi. Triple systems with no six points carrying three triangles. *Combinatorics (Keszthely, 1976), Coll. Math. Soc. J. Bolyai*, 18:939–945, 1978.
- 34 I. D. Shkredov. On a two-dimensional analogue of Szemerédi's theorem in Abelian groups. *Izvestiya: Mathematics*, 73(5):1033–1075, 2009.
- 35 A. Shraibman. A Note on Multiparty Communication Complexity and the Hales-Jewett Theorem. *arXiv preprint*, 2017. [arXiv:1706.02277](https://arxiv.org/abs/1706.02277).
- 36 J. Solymosi. Note on a Generalization of Roth's Theorem. In *Discrete and Computational Geometry: The Goodman-Pollack Festschrift*, pages 825–827. Springer Berlin Heidelberg, Berlin, Heidelberg, 2003.
- 37 P. Tesson. An application of the Hales-Jewett Theorem to multiparty communication complexity, 2004.
- 38 L. Trevisan. Guest column: additive combinatorics and theoretical computer science. *SIGACT News*, 40(2):50–66, 2009.
- 39 J. H. van Lint and R. M. Wilson. *A course in combinatorics*. Cambridge university press, 2001.
- 40 A. Yao. On ACC and threshold circuits. In *Proceedings of the 31st IEEE Symposium on Foundations of Computer Science*, pages 619–627. IEEE, 1990.



# Fisher Zeros and Correlation Decay in the Ising Model

Jingcheng Liu<sup>1</sup>

Computer Science Division, UC Berkeley, USA  
liuexp@berkeley.edu

Alistair Sinclair<sup>2</sup>

Computer Science Division, UC Berkeley, USA  
sinclair@cs.berkeley.edu

Piyush Srivastava<sup>3</sup>

Tata Institute of Fundamental Research, Mumbai, India  
piyush.srivastava@tifr.res.in

---

## Abstract

The Ising model originated in statistical physics as a means of studying phase transitions in magnets, and has been the object of intensive study for almost a century. Combinatorially, it can be viewed as a natural distribution over cuts in a graph, and it has also been widely studied in computer science, especially in the context of approximate counting and sampling. In this paper, we study the complex zeros of the partition function of the Ising model, viewed as a polynomial in the “interaction parameter”; these are known as *Fisher zeros* in light of their introduction by Fisher in 1965. While the zeros of the partition function as a polynomial in the “field” parameter have been extensively studied since the classical work of Lee and Yang, comparatively little is known about Fisher zeros. Our main result shows that the zero-field Ising model has no Fisher zeros in a complex neighborhood of the entire region of parameters where the model exhibits correlation decay. In addition to shedding light on Fisher zeros themselves, this result also establishes a formal connection between two distinct notions of phase transition for the Ising model: the absence of complex zeros (analyticity of the free energy, or the logarithm of the partition function) and decay of correlations with distance. We also discuss the consequences of our result for efficient deterministic approximation of the partition function. Our proof relies heavily on algorithmic techniques, notably Weitz’s self-avoiding walk tree, and as such belongs to a growing body of work that uses algorithmic methods to resolve classical questions in statistical physics.

**2012 ACM Subject Classification** Mathematics of computing → Discrete mathematics

**Keywords and phrases** Ising model, zeros of polynomials, partition functions, approximate counting, phase transitions

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2019.55

**Related Version** Full version available at [15], <https://arxiv.org/abs/1807.06577>.

---

<sup>1</sup> Supported by US NSF grant CCF-1815328.

<sup>2</sup> Supported by US NSF grant CCF-1815328.

<sup>3</sup> Supported by a Ramanujan Fellowship of the Indian Department of Science and Technology.



© Jingcheng Liu, Alistair Sinclair, and Piyush Srivastava;  
licensed under Creative Commons License CC-BY

10th Innovations in Theoretical Computer Science (ITCS 2019).

Editor: Avrim Blum; Article No. 55; pp. 55:1–55:8

Leibniz International Proceedings in Informatics



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

## 1 Introduction

In combinatorial terms, the Ising model is a probability distribution over the cuts of a graph. Given a graph  $G = (V, E)$ , the configurations of the model are assignments  $\sigma$  of “+” or “−” spins to the vertices of  $G$ ;  $\sigma$  corresponds to the cut between spin-“+” and spin-“−” vertices. The model assigns to configuration  $\sigma$  the weight  $w_{G,\beta}(\sigma) = \beta^{|\{e=(u,v) \in E : \sigma(u) \neq \sigma(v)\}|}$ , where  $\beta > 0$  is a parameter. The associated probability distribution, known as the *Gibbs measure*, is then defined by assigning probability  $\mu_{G,\beta}(\sigma) := \frac{1}{Z_G(\beta)} w_{G,\beta}(\sigma)$  to each configuration  $\sigma$ . The normalizing factor here is the *partition function*, defined as

$$Z_G(\beta) := \sum_{\sigma: V \rightarrow \{+, -\}} w_{G,\beta}(\sigma) = \sum_{k=0}^{|E|} \gamma_k \beta^k, \quad (1)$$

where  $\gamma_k$  is the number of  $k$ -edge cuts in  $G$ . Note that  $Z_G(\beta)$  is a polynomial in  $\beta$  with positive coefficients. We also sometimes consider graphs in which certain vertices are *pinned* to “+” or “−” spins. For such a graph, we restrict the sum in the definition of  $Z_G$  to those configurations  $\sigma$  in which these vertices have the spin determined by their pinning.

The origins of the Ising model lie in the qualitative modeling of phase transitions in magnets [11]; indeed, it was the first model among the wide class of *spin systems* to be studied extensively in statistical physics. The parameter  $\beta$  above is a proxy for the “temperature” or “interaction strength”, while the graph is a proxy for the physical structure of the magnet. In this parameterization,  $\beta > 1$  corresponds to so-called anti-ferromagnetic interactions (where neighbors prefer to have different spins),  $\beta < 1$  to ferromagnetic interactions (where neighbors prefer to have the same spins), and  $\beta = 1$  to infinite temperature (where the neighbors behave independently of each other). We will restrict our attention throughout to graphs of fixed (but arbitrary) maximum degree  $\Delta$ .

Historically, there have been two distinct (though closely related) mechanisms for defining and understanding phase transitions in statistical physics. The first is decay of long-range correlations in the Gibbs measure, which is familiar in theoretical computer science due to its extensive use in approximation algorithms and the analysis of spin systems and graphical models. The second, which is more classical and less familiar in computer science, is analyticity of the “free energy”  $\log Z$  (where  $Z$  is the partition function). This second notion connects naturally to stability theory of polynomials, and in particular to the study of the location of *complex* roots of the partition function  $Z$ , even when only real values of the parameters make physical sense in the model. The seminal work of Lee and Yang [12, 28] was one of the first, and certainly the best known, to use this notion. We note in passing that stability theory has seen a surge of recent interest in theoretical computer science, in contexts ranging from approximation algorithms to the construction of Ramanujan graphs (see, e.g., [17, 18, 1, 2, 26]).

We now briefly describe the connection between the analyticity of the free energy and the location of complex zeros of the partition function. The first ingredient is that natural observables of the model (e.g., the *magnetization*) can be written as derivatives of the free energy with respect to an appropriate parameter of the model. Thus, analyticity of the free energy for a given range  $S$  of parameters implies that all such observables vary continuously (and have continuous derivatives) when the parameter value lies in  $S$ , which in turn implies that there is no phase transition in  $S$ . However, it is not hard to see that for any finite graph, the free energy is always analytic as a function of  $\beta$  when  $\beta$  lies on the positive real axis, suggesting a complete absence of phase transitions. Indeed, it turns out (see, e.g., [23, Chapter 1]) that in order to see phase transitions one has to consider *infinite* graphs.

For concreteness, we consider the case of the Ising model on the infinite 2-dimensional integer lattice  $\mathbb{Z}^2$  [28]. In order to define the free energy for such an infinite graph, one takes the limit of the free energies of a suitable increasing sequence of finite subgraphs (e.g., increasing rectangles in  $\mathbb{Z}^2$ ), after scaling them by their size. Lee and Yang [28] showed that, for infinite graphs of sub-exponential growth (including  $\mathbb{Z}^2$ ), the free energy obtained via this prescription is well defined and analytic for a range of parameters  $S$  provided that the partition functions of the finite graphs used in the limit definition, viewed as polynomials in the parameter, are zero-free in a *complex* neighborhood of  $S$ . Thus, proving zero-freeness of partition functions of such a sequence of finite graphs in a *fixed* (i.e., not depending upon the finite graphs in question) complex neighborhood of  $S$  implies the absence of phase transitions in  $S$ .

**Algorithms, phase transitions, and roots of polynomials.** While the algorithmic consequences of phase transitions defined in terms of decay of correlations have been well studied, first in the context of Markov Chain Monte Carlo algorithms (Glauber dynamics) and more recently in deterministic algorithms that directly exploit correlation decay, algorithmic use of the information on complex roots of the partition function originated only recently in the work of Barvinok (see [3] for a survey). This has led to increased interest in understanding the relationship between the above two notions of phase transitions. Such connections have been the focus of some recent work on the independent set (or “hard core lattice gas”) model; notably, connections similar to the ones in this paper have been explored for that model by Peters and Regts [20], while related ideas are harnessed in early work of Shearer [22], as later elucidated by Scott and Sokal [21] and further elaborated by Harvey *et al.* [10], to shed light on the Lovász Local Lemma.

The motivation for our work here is to take a step towards achieving a fuller understanding of these connections. Specifically, we study the zeros of the Ising partition function (at zero field), viewed as a polynomial in the interaction parameter. While the study of zeros in terms of the fugacity (or field) parameter was famously pioneered by Lee and Yang [12], and has given rise to a well developed theory, very little is known about the zeros in terms of the interaction parameter, which were first studied in the classical 1965 paper of Fisher [8] and are thus known as “Fisher zeros”.

Our main result is that the Ising model has no Fisher zeros in a region of the complex plane that contains the entire interval  $B$  on the positive real line where correlation decay holds. Our analysis crucially exploits the correlation decay property in order to understand the Fisher zeros. Thus, in the particular case of the zero field Ising model, we are able to establish a tight connection between correlation decay and the absence of zeros. Another potentially interesting aspect of this result is the use of algorithmic techniques associated with correlation decay (notably, Weitz’s algorithm [27]) to understand a classical concept in statistical physics.

We now proceed to formally describe our results. First we identify the range of the parameter  $\beta$  for which the Ising model is, in a certain sense, well-behaved on graphs of bounded degree  $\Delta$ .

► **Definition 1** (Correlation decay region). Given  $\Delta > 0$ , the *correlation decay region*  $B = B_\Delta$  for  $\beta$  is the interval  $(\frac{\Delta-2}{\Delta}, \frac{\Delta}{\Delta-2})$ .

The correlation decay region is very well studied in both physical and algorithmic contexts, and comes from a consideration of the behavior of the Gibbs measure on trees. In particular, it corresponds to those  $\beta$  for which there is exponential decay of correlations in the Gibbs measure on any finite subtree of the infinite  $\Delta$ -regular tree – a fact which has been used to

give a deterministic algorithm for approximating the partition function of the Ising model for such  $\beta$  [27, 29]. On the other hand, Sly and Sun [25] have shown that for  $\beta > \frac{\Delta}{\Delta-2}$ , this approximation problem is NP-hard under randomized reductions. In statistical physics, the correlation decay region describes those  $\beta$  for which the definition of the Gibbs measure given by eq. (1) for finite graphs can be extended in a unique way to a Gibbs measure on the *infinite*  $\Delta$ -regular tree [9]; for this reason, the correlation decay region is also referred to as the *uniqueness region*.

As advertised earlier, our goal is to prove the existence of a region of the complex plane, containing  $B$ , which contains no Fisher zeros. We state this now as our main theorem.

► **Theorem 2.** *Fix any  $\Delta > 0$ . For any real  $\beta \in B := (\frac{\Delta-2}{\Delta}, \frac{\Delta}{\Delta-2})$ , there exists a  $\delta > 0$  such that for all  $\beta' \in \mathbb{C}$  with  $|\beta' - \beta| < \delta$ , the Ising partition function  $Z_G(\beta') \neq 0$  for all graphs  $G$  of maximum degree  $\Delta$ . Moreover the same holds even if  $G$  contains an arbitrary number of vertices pinned to  $+$  or  $-$  spins.*

► **Remarks.**

- (1) It is worth noting that the choice of  $\delta$  does not depend on the size of the graph, only on  $\Delta$  and  $\beta$ . In particular, given any  $\delta_1 > 0$ , one can choose  $\delta > 0$  such that, for all  $\beta'$  in a complex neighborhood of radius  $\delta$  around the closed interval  $[\frac{\Delta-2}{\Delta} + \delta_1, \frac{\Delta}{\Delta-2} - \delta_1]$ ,  $Z_G(\beta')$  is non-zero for all graphs of degree at most  $\Delta$ .
- (2) For the case of the Ising model, the above theorem establishes a connection between the two notions of phase transition discussed above. Namely, for the zero-field Ising model, it shows that decay of correlations on the  $\Delta$ -regular tree also implies the absence of Fisher zeros for finite graphs of degree at most  $\Delta$ , and hence the analyticity of the free energy for appropriate infinite graphs (i.e., those of maximum degree at most  $\Delta$  and of subexponential growth, such as regular lattices).

**Discussion.** To the best of our knowledge, the previous best general result on the Fisher zeros of the Ising model appears in the work of Barvinok and Soberón [6], who showed that  $Z_G(\beta)$  is non-zero if  $|\beta - 1| < c/\Delta$ , where  $\Delta$  is the maximum degree of  $G$ , and  $c$  can be chosen to be 0.34 (and as large as 0.45 if  $\Delta$  is large enough). While this result provides a disk around 1 in which there are no Fisher zeros, it cannot guarantee the absence of Fisher zeros in a neighborhood of the correlation decay region  $B$  (which would require at least that  $c \geq 2 - o_\Delta(1)$ ). Our Theorem 2 therefore strengthens this result to a neighborhood of the entire correlation decay region  $B$ .<sup>4</sup>

Our main theorem on Fisher zeros can also be combined with the techniques of Barvinok [3] and Patel and Regts [19] to give a new deterministic polynomial time approximation algorithm for the partition function of the ferromagnetic Ising model with zero field on graphs of degree at most  $\Delta$  when  $\beta \in (\frac{\Delta-2}{\Delta}, \frac{\Delta}{\Delta-2})$ . In particular, combining Theorem 2 with Lemmas 2.2.1 and 2.2.3 of [3] (see also the discussion at the bottom of page 27 therein) and the proof of Theorem 6.1 of [19], we obtain the following corollary:

► **Corollary 3.** *Fix a positive integer  $\Delta$  and  $\delta > 0$ . There exist positive constants  $\delta_1 > 0$  and  $c$  such that for any complex  $\beta$  with  $\Re(\beta) \in [\frac{\Delta-2}{\Delta} + \delta, \frac{\Delta}{\Delta-2} - \delta]$  and  $|\Im(\beta)| \leq \delta_1$ , the following is true. There exists an algorithm which, on input a graph  $G$  of degree at most  $\Delta$  on  $n$  vertices, and an accuracy parameter  $\varepsilon > 0$ , runs in time  $O(n/\varepsilon)^c$  and outputs  $\hat{Z}$  satisfying  $|\hat{Z} - Z_G(\beta)| \leq \varepsilon |Z_G(\beta)|$ .*

<sup>4</sup> Technically the results are incomparable in the sense that, while our results cover a much larger portion of the real line than that in [6], the diameter of the disk centered around 1 in the region of [6] may be larger than the radius guaranteed by our result.

For real  $\beta$  in the same range, a deterministic algorithm with the above properties, based on correlation decay, was already analyzed in [29]. However, our extension to complex values of the parameter is also interesting in light of the fact that algorithms for approximating the Ising partition function at complex values of the parameters have recently been studied in the context of classical simulation of restricted models of quantum computation [16].

Finally, we emphasize that in contrast to most other recent applications of Barvinok’s method (e.g., [19, 5, 6, 4, 14]), where the required results on the location of the roots of the associated partition function are derived without reference to correlation decay, the algorithmic version of correlation decay is crucial to our proof. Indeed, implicit in our proof is an analysis of Weitz’s celebrated correlation decay algorithm [27] (proposed originally for the independent set (or “hard core”) model, and analyzed by Zhang, Liang and Bai [29] for the Ising model in the case of real positive  $\beta \in B$ ) for the Ising model with *complex*  $\beta'$  close to  $\beta \in B$ . Thus, as mentioned earlier, our work shows that Weitz’s algorithm can be viewed as a bridge between the “decay of correlations” and “analyticity of free energy” views of phase transitions. We note also that our work is close in spirit to recent work of Peters and Regts [20] (see also [7]), who employ correlation decay in the hard core model to prove stability results for the hard core partition function.

## 2 Outline of proof

We fix  $\Delta$  to be the maximum degree throughout, and let  $d = \Delta - 1$ . Let  $G$  be any graph of maximum degree  $\Delta$ . Our starting point is a recursive criterion that guarantees that the partition function  $Z_G(\beta)$  has no zeros. For any non-isolated vertex  $v$  of  $G$ , let  $Z_{G,v}^+(\beta)$  (respectively,  $Z_{G,v}^-(\beta)$ ) be the contribution to  $Z_G(\beta)$  from configurations with  $\sigma(v) = +$  (respectively, with  $\sigma(v) = -$ ), so that  $Z_G(\beta) = Z_{G,v}^+(\beta) + Z_{G,v}^-(\beta)$ . Define also the ratio  $R_{G,v}(\beta) := \frac{Z_{G,v}^+(\beta)}{Z_{G,v}^-(\beta)}$ . Now note that  $Z_{G,v}^+(\beta)$  and  $Z_{G,v}^-(\beta)$  can be seen as Ising partition functions defined on the same graph  $G$  with the vertex  $v$  *pinned* to the appropriate spin; i.e., they are partition functions defined on a graph with one less unpinned vertex. Thus we may assume recursively that neither  $Z_{G,v}^+(\beta)$  nor  $Z_{G,v}^-(\beta)$  vanishes. Under this assumption, the condition  $Z_G(\beta) \neq 0$  is equivalent to  $R_{G,v}(\beta) \neq -1$ .

Our next ingredient is a formal recurrence, due to Weitz [27], for computing ratios such as  $R_{G,v}(\beta)$  in two-state spin systems. This recurrence is based on the so-called “tree of self-avoiding walks” (or “SAW tree”) in  $G$ , rooted at  $v$ , with appropriate boundary conditions (i.e., initial inputs, or fixed values at the leaves of the tree). Weitz’s recurrence has been used in the development of several approximate counting algorithms based on decay of correlations (see, e.g., [27, 29, 13, 24]). We now state a precise version of Weitz’s result that is tailored to our application.

► **Lemma 4.** *Let  $G$  be a graph of maximum degree  $\Delta = d + 1$ , with some vertices possibly pinned to spins “+” or “−”. Given  $\beta \in \mathbb{C}$ , define  $h_\beta(x) := \frac{\beta+x}{\beta x+1}$ . For integers  $k \geq 0$  and  $s$ , define the maps*

$$F_{\beta,k,s}(\mathbf{x}) := \beta^s \prod_{i=1}^k h_\beta(x_i).$$

*Then, the ratio  $R_{G,v}(\beta)$  can be obtained by iteratively applying a sequence of multivariate maps of the form  $F_{\beta,k,s}(\mathbf{x})$  such that, in all but the final application, one has  $1 \leq k + |s| \leq d$ , while for the final application one has  $1 \leq k + |s| \leq \Delta$ , and any initial input to these maps is  $x_i = 1$ .*

For completeness we sketch a proof of Lemma 4 at the end of this section.

Returning now to the condition  $R_{G,v}(\beta) \neq -1$  derived above, we see from Lemma 4 that a sufficient condition for the absence of zeros of  $Z_G(\beta)$  is the existence of a subset  $D \subseteq \mathbb{C}$  such that  $1 \in D$ ,  $-1 \notin D$ , and  $D$  is closed under the recurrence  $F_{\beta,k,s}$  (in the sense that  $F_{\beta,k,s}$  maps  $D^k$  into  $D$ ). These properties guarantee that the recurrence, with initial inputs 1 at the leaves, can never yield the value  $-1$ , and hence that  $R_{G,v}(\beta) \neq -1$ , so  $Z_G(\beta) \neq 0$ . The main technical content of this paper is to prove, under the conditions on  $\beta$  stated in Theorem 2, the existence of such a set  $D$ , a result which we formally state as follows.

► **Theorem 5.** *Fix a degree  $\Delta = d + 1$ . For any  $\beta \in (\frac{\Delta-2}{\Delta}, \frac{\Delta}{\Delta-2})$ , there exists  $\delta_\beta > 0$  such that, for any  $\beta' \in \mathbb{C}$  with  $|\beta' - \beta| \leq \delta_\beta$ , there exists a set  $D \subseteq \mathbb{C}$  with  $1 \in D$ ,  $-1 \notin D$ , and*

- (a)  $F_{\beta',k,s}(D^k) \subseteq D$  for integers  $k \geq 0$  and  $s$  such that  $1 \leq k + |s| \leq d$ ;
- (b)  $-1 \notin F_{\beta',k,s}(D^k)$  for integers  $k \geq 0$  and  $s$  such that  $1 \leq k + |s| \leq \Delta$ .

At the end of this section, we spell out the details of how to combine Lemma 4 and Theorem 5 into a proof of our main result, Theorem 2.

The main technical task of the paper is to prove Theorem 5. We briefly sketch our approach in this extended abstract; the details can be found in the full version [15]. The first step is to simplify the problem by working with a *univariate* version of the recurrence  $F_{\beta,k,s}$  defined in Lemma 4. The univariate version is defined as  $f_{\beta,k,s}(x) := \beta^s h_\beta(x)^k$ , and we can show that it satisfies  $F_\beta(D^k) = f_\beta(D)$  for any set  $D$  such that  $C := \log(h_\beta(D))$  is convex in the complex plane. (Henceforth we will drop the subscripts  $k, s$  for simplicity.) This means that the set  $D$  we seek in Theorem 5 should be the image of a convex set  $C$  under the map  $\log \circ h_\beta$ .

Next, to enable us to exploit the fact that  $\beta$  is in the correlation decay interval  $B = (\frac{\Delta-2}{\Delta}, \frac{\Delta}{\Delta-2})$ , we further modify the univariate recurrence to  $f_\beta^\varphi := \varphi \circ f_\beta \circ \varphi^{-1}$ , where  $\varphi(x) := \log x$ . This is an example of the use of a so-called “potential” function  $\varphi$  in order to smooth a recurrence, as has been useful in several correlation decay arguments. The key point here is that, when  $\beta \in B$ ,  $f_\beta^\varphi$  (unlike  $f_\beta$  itself) is actually a uniform *contraction* on an appropriate domain in  $\mathbb{C}$ ; hence we can conclude that  $f_\beta^\varphi(S) \subseteq S$  for “nice” sets  $S$  (i.e.,  $S$  that are convex and symmetric around the origin). Since the condition  $f_\beta(D) \subseteq D$  is equivalent to  $f_\beta^\varphi(\log D) \subseteq \log D$ , this imposes the further constraint that  $\log D$  be a “nice” set.

Putting together the constraints in the previous two paragraphs, we need to construct a suitable convex set  $C$  whose image  $\log(h_\beta^{-1}(\exp(C)))$  is nice; our set  $D$  in Theorem 5 will then be defined as  $h_\beta^{-1}(\exp(C))$  (and this set must include 1 and exclude  $-1$ ). This turns out to be hard to achieve directly due to the complexity of the map  $p := \log \circ h_\beta^{-1} \circ \exp$ . However, we are able to show that one can instead work with a (non-analytic) approximation of  $p$  under which the image of a natural convex  $C$  becomes a nice (in fact, rectangular) set. Moreover, this holds even for complex  $\beta$  that are sufficiently close to the region  $B$ . This fact then allows us to push through the analysis and arrive at a proof of Theorem 5.

We conclude this overview with the proofs of Lemma 4 and Theorem 2 promised earlier. The proof of our main technical result, Theorem 5, can be found in the full version [15].

**Proof of Lemma 4 (Sketch).** This description is exactly the same as the version of Weitz’s result used for the Ising model in, e.g., [24] and [29], except that there, only the maps  $F_{\beta,k,0}$  for  $1 \leq k \leq d$  (with at most one final application with  $k = \Delta$ ) are used, and the initial values come from the set  $\{0, \infty\}$ ; these initial values are the values of the ratio for single leaf vertices in the SAW tree pinned to  $-$  and  $+$  respectively, and the maps  $F_{\beta,k,0}$  describe how to combine the ratios from  $k$  subtrees. The version in the lemma follows by



noticing that  $h_\beta(1) = 1, h_\beta(0) = \beta$  and  $h_\beta(\infty) = 1/\beta$ , so that  $F_{\beta,k,0}$  applied to a vector  $\mathbf{x}$  with  $k$  coordinates,  $s_1$  of which are set to 0 and  $s_2$  to  $\infty$ , produces the same output as  $F_{\beta,k-|s_1-s_2|,s_1-s_2}$  applied to the vector  $\mathbf{x}'$  of  $k - |s_1 - s_2|$  coordinates obtained from  $\mathbf{x}$  by removing the 0 and  $\infty$  entries, and then appending  $s_1 + s_2 - |s_1 - s_2|$  entries which are 1. ◀

**Proof of Theorem 2.** As indicated earlier, the induction is on the number of unpinned vertices,  $n$ , of  $G$ . For the base case  $n = 0$ ,  $Z_G(\beta) = \beta^k$ , where  $k$  is the number of pairs of adjacent vertices in  $G$  that are pinned to different spins. Therefore,  $Z_G(\beta) \neq 0$  unless  $\beta = 0$ . Next suppose that for some positive integer  $t$ , it holds that for every  $\beta \in B$ , there exists a  $\delta > 0$  such that for all  $\beta' \in \mathbb{C}$  with  $|\beta' - \beta| < \delta$ ,  $Z_G(\beta') \neq 0$  for all graphs  $G$  of maximum degree  $\Delta$  with at most  $t$  unpinned vertices. Now, let  $G'$  be any graph of the same maximum degree with  $t + 1$  unpinned vertices. Fix any non-isolated vertex  $v$  in  $G'$ , and let  $Z_{G',v}^+(\beta'), Z_{G',v}^-(\beta')$  be the contributions to the partition function from configurations with  $\sigma(v) = +, \sigma(v) = -$ , respectively. By the induction hypothesis, we know that  $Z_{G',v}^+(\beta') \neq 0, Z_{G',v}^-(\beta') \neq 0$  as they are exactly the Ising partition function defined on the same graph  $G'$  with the vertex  $v$  pinned (thus reducing the number of unpinned vertices to  $t$ ). Further, Lemma 4 implies that  $R_{G',v}(\beta') = \frac{Z_{G',v}^+(\beta')}{Z_{G',v}^-(\beta')}$  can be computed by iteratively applying a sequence of maps of the form  $F_{\beta',k,s}$  for  $1 \leq k + |s| \leq d$ , followed by at most one application where  $k + |s| = \Delta$ , starting with initial values of 1. Part (a) of Theorem 5 then implies that the outputs of all except possibly the final application remain in the set  $D$  defined in that theorem, and part (b) of the theorem implies that the final output, which is equal to  $R_{G',v}(\beta)$  by Lemma 4, is not  $-1$ . Since  $Z_{G',v}^+(\beta')$  and  $Z_{G',v}^-(\beta')$  are non-zero, this implies that  $Z_{G'}(\beta') \neq 0$ , completing the induction. ◀

---

## References

- 1 Nima Anari and Shayan Oveis Gharan. The Kadison-Singer problem for strongly Rayleigh measures and applications to Asymmetric TSP. In *Proc. 56th IEEE Symp. Found. Comp. Sci. (FOCS)*, October 2015.
- 2 Nima Anari and Shayan Oveis Gharan. A Generalization of Permanent Inequalities and Applications in Counting and Optimization. In *Proc. 49th ACM Symp. Theory Comput. (STOC)*, pages 384–396. ACM, June 2017. arXiv:1702.02937.
- 3 A. Barvinok. *Combinatorics and Complexity of Partition Functions*. Algorithms and Combinatorics. Springer, 2017.
- 4 Alexander Barvinok. Computing the permanent of (some) complex matrices. *Found. Comput. Math.*, 16(2):329–342, 2015. doi:10.1007/s10208-014-9243-7.
- 5 Alexander Barvinok and Pablo Soberón. Computing the partition function for graph homomorphisms with multiplicities. *J. Combin. Theory, Ser. A*, 137:1–26, 2016.
- 6 Alexander Barvinok and Pablo Soberón. Computing the partition function for graph homomorphisms. *Combinatorica*, 37(4):633–650, August 2017.
- 7 Ferenc Bencs and Péter Csikvári. Note on the zero-free region of the hard-core model, July 2018. arXiv:1807.08963.
- 8 M. E. Fisher. The nature of critical points. In W. E. Brittin, editor, *Lecture notes in Theoretical Physics*, volume 7c, pages 1–159. University of Colorado Press, 1965.
- 9 Hans-Otto Georgii. *Gibbs Measures and Phase Transitions*. De Gruyter Studies in Mathematics. Walter de Gruyter Inc., October 1988.
- 10 Nicholas J. A. Harvey, Piyush Srivastava, and Jan Vondrák. Computing the independence polynomial: from the tree threshold down to the roots. In *Proc. 29th ACM-SIAM Symp. Discrete Algorithms (SODA)*, pages 1557–1576, 2018. arXiv:1608.02282.



- 11 E. Ising. Beitrag zur Theorie des Ferromagnetismus. *Z. Phys.*, 31:253–258, February 1925. doi:10.1007/BF02980577.
- 12 T. D. Lee and C. N. Yang. Statistical Theory of Equations of State and Phase Transitions. II. Lattice Gas and Ising Model. *Phys. Rev.*, 87(3):410–419, 1952. doi:10.1103/PhysRev.87.410.
- 13 Liang Li, Pinyan Lu, and Yitong Yin. Correlation Decay up to Uniqueness in Spin Systems. In *Proc. 24th ACM-SIAM Symp. Discrete Algorithms (SODA)*, pages 67–84, 2013.
- 14 Jingcheng Liu, Alistair Sinclair, and Piyush Srivastava. The Ising Partition Function: Zeros and Deterministic Approximation. In *Proc. 58th Annual IEEE Symp. Found. Comp. Sci. (FOCS)*, pages 986–997, 2017. arXiv:1704.06493.
- 15 Jingcheng Liu, Alistair Sinclair, and Piyush Srivastava. Fisher zeros and correlation decay in the Ising model, 2018. arXiv:1807.06577.
- 16 Ryan L. Mann and Michael J. Bremner. Approximation algorithms for complex-valued Ising models on bounded degree graphs, June 2018. arXiv:1806.11282.
- 17 Adam Marcus, Daniel Spielman, and Nikhil Srivastava. Interlacing families I: Bipartite Ramanujan graphs of all degrees. *Ann. Math.*, pages 307–325, July 2015. doi:10.4007/annals.2015.182.1.7.
- 18 Adam W Marcus, Daniel A Spielman, and Nikhil Srivastava. Interlacing families II: Mixed characteristic polynomials and the Kadison-Singer problem. *Ann. Math.*, 182:327–350, 2015.
- 19 Viresh Patel and Guus Regts. Deterministic polynomial-time approximation algorithms for partition functions and graph polynomials. *SIAM J. Comput.*, 46(6):1893–1919, December 2017. arXiv:1607.01167.
- 20 Han Peters and Guus Regts. On a conjecture of Sokal concerning roots of the independence polynomial, January 2017. arXiv:1701.08049.
- 21 Alex Scott and Alan Sokal. The repulsive lattice gas, the independent-set polynomial, and the Lovász local lemma. *J. Stat. Phys.*, 118(5-6):1151–1261, 2004.
- 22 James B. Shearer. On a problem of Spencer. *Combinatorica*, 5(3), 1985.
- 23 Barry Simon. *The Statistical Mechanics of Lattice Gases*, volume 1. Princeton University Press, 1993.
- 24 Alistair Sinclair, Piyush Srivastava, and Marc Thurley. Approximation Algorithms for Two-State Anti-Ferromagnetic Spin Systems on Bounded Degree Graphs. *J. Stat. Phys.*, 155(4):666–686, 2014.
- 25 Allan Sly and Nike Sun. Counting in two-spin models on  $d$ -regular graphs. *Ann. Probab.*, 42(6):2383–2416, November 2014. doi:10.1214/13-AOP888.
- 26 Damian Straszak and Nisheeth K. Vishnoi. Real Stable Polynomials and Matroids: Optimization and Counting. In *Proc. 49th ACM Symp. Theory Comput. (STOC)*, pages 370–383. ACM, June 2017. arXiv:1611.04548.
- 27 Dror Weitz. Counting Independent Sets up to the Tree Threshold. In *Proc. 38th ACM Symp. Theory Comput. (STOC)*, pages 140–149, 2006.
- 28 C. N. Yang and T. D. Lee. Statistical Theory of Equations of State and Phase Transitions. I. Theory of Condensation. *Phys. Rev.*, 87(3):404–409, August 1952. doi:10.1103/PhysRev.87.404.
- 29 Jinshan Zhang, Heng Liang, and Fengshan Bai. Approximating partition functions of the two-state spin system. *Inf. Process. Lett.*, 111(14):702–710, 2011. doi:10.1016/j.ipl.2011.04.012.


# Quadratic Time-Space Lower Bounds for Computing Natural Functions with a Random Oracle

Dylan M. McKay

EECS and CSAIL, MIT, 32 Vassar St., Cambridge MA, USA  
dmmckay@mit.edu

Richard Ryan Williams<sup>1</sup>

EECS and CSAIL, MIT, 32 Vassar St., Cambridge MA, USA  
rrw@mit.edu

 <https://orcid.org/0000-0003-2326-2233>

---

## Abstract

We define a model of size- $S$   $R$ -way branching programs with oracles that can make up to  $S$  distinct oracle queries over all of their possible inputs, and generalize a lower bound proof strategy of Beame [SICOMP 1991] to apply in the case of random oracles. Through a series of succinct reductions, we prove that the following problems require randomized algorithms where the product of running time and space usage must be  $\Omega(n^2/\text{poly}(\log n))$  to obtain correct answers with constant nonzero probability, even for algorithms with constant-time access to a uniform random oracle (i.e., a uniform random hash function):

- Given an unordered list  $L$  of  $n$  elements from  $[n]$  (possibly with repeated elements), output  $[n] - L$ .
- Counting satisfying assignments to a given 2CNF, and printing any satisfying assignment to a given 3CNF. Note it is a major open problem to prove a time-space product lower bound of  $n^{2-o(1)}$  for the *decision version* of SAT, or even for the decision problem Majority-SAT.
- Printing the truth table of a given CNF formula  $F$  with  $k$  inputs and  $n = O(2^k)$  clauses, with values printed in lexicographical order (i.e.,  $F(0^k), F(0^{k-1}1), \dots, F(1^k)$ ). Thus we have a  $4^k/\text{poly}(k)$  lower bound in this case.
- Evaluating a circuit with  $n$  inputs and  $O(n)$  outputs.

As our lower bounds are based on  $R$ -way branching programs, they hold for any reasonable model of computation (e.g. log-word RAMs and multitape Turing machines).

**2012 ACM Subject Classification** Theory of computation → Circuit complexity, Theory of computation → Oracles and decision trees

**Keywords and phrases** branching programs, random oracles, time-space tradeoffs, lower bounds, SAT, counting complexity

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2019.56

**Funding** Supported by NSF CCF-1741615 (CAREER: Common Links in Algorithms and Complexity).

**Acknowledgements** We thank the anonymous reviewers for helpful comments.

---

<sup>1</sup> Parts of this work were performed while visiting the Simons Institute for the Theory of Computing and the EECS department at UC Berkeley.



## 1 Introduction

Infamously little progress has been made towards the resolution of P vs NP. It is still open whether there are linear time algorithms for solving generic NP-hard problems such as CIRCUIT SAT although for certain NP-hard problems, non-linear lower bounds on multitape Turing machines are known [23, 18]. In fact it remains open whether CIRCUIT SAT has a generic algorithm in a random-access machine model running in  $O(n^{1.9999})$  time and  $O(\log n)$  additional space beyond the input.<sup>2</sup> Currently the best known time lower bound for solving SAT in  $O(\log n)$  space is  $n^{2 \cos(\pi/7) - o(1)} \geq n^{1.801}$  ([27, 14], building on [17]).

Other prominent work has used combinatorial methods to prove time-space lower bounds for decision problems in P [10, 3, 12, 11, 25, 4, 22]. For general random-access models of computation, modest super-linear (e.g.  $n \log n$ -type) time lower bounds for explicit problems in P are known when the space is restricted to  $n^{.99}$  or less (for randomized models as well [11, 22]). Thus for such problems, the time-space product can be lower-bounded to nearly  $\Omega(n^2)$  when the space is close to  $n$ ; however, when the space is  $O(\log n)$ , the best time lower bound against RAMs appears to be the aforementioned  $n^{1.801}$  bound for SAT.

All the above cited lower bounds are for *decision* problems. For example, the SAT lower bound applies to algorithms which are only required to determine if a given formula is satisfiable or not. There are several well-studied extensions of SAT which are *function* problems, outputting multiple bits:

1. **How difficult is it to *print* a satisfying assignment, when one exists?** For 3CNF formulas, call this problem PRINT-3SAT. Of course, PRINT-3SAT has a polynomial-time algorithm if and only if P = NP, but perhaps it is easier to prove concrete lower bounds for it, compared to the decision version.
2. **How difficult is #2SAT, in which we *count* the number of satisfying assignments to a 2CNF?** Since #2SAT is #P-complete, the problem should intuitively be harder to solve than 3SAT.
3. **How difficult is it to print the truth table of a CNF?** Given a CNF formula  $F$  of  $n$  variables and up to  $2^n$  size,  $F$  can be evaluated on all possible  $2^n$  inputs in  $4^n \cdot \text{poly}(n)$  time and  $\text{poly}(n)$  space (using a linear-time, log-space algorithm for evaluating a CNF on a given input). Call this problem TTPRINT (for truth-table printing).

**Is this quadratic running time optimal?** This question is considerable interest for SAT algorithms; for some circuit classes, the only known SAT algorithms beating exhaustive search (e.g., [28]) proceed by reducing SAT to a quick truth table evaluation.

In this paper, we prove that for essentially any generic randomized computation model using  $n^{o(1)}$  space but having  $O(1)$ -access to a random oracle (i.e., a random string of  $2^{n^{o(1)}}$  bits), the above three problems all require nearly quadratic time to compute with nonzero constant probability. We revisit a quadratic time-space lower bound of Beame [8] for the UNIQUE ELEMENTS problem, show that his technique can be used to prove an analogous lower bound for an even “simpler” problem that we call NON-OCCURRING ELEMENTS, extend the lower bound’s reach to include random oracles, and give succinct reductions from NON-OCCURRING ELEMENTS to the above problems, proving hardness for them.

<sup>2</sup> Note that for multitape Turing machines, such lower bounds are not hard to show [24]. However, these lower bounds rely on the sequential access of Turing machines on tapes; once random access is allowed, these lower bounds break.

### Lower Bound for Non-Occurring Elements

We define the NON-OCCURRING ELEMENTS problem to be: *given an unordered list  $L$  of  $n$  elements from  $[n]$ , output  $[n] - L$  in any order.*<sup>3</sup> That is, we simply wish to output the elements that do not occur in the given list.

Observe it is easy to get a space- $O(s(n))$  algorithm on the  $\log(n)$ -word RAM for computing NOE with running time  $O(n^2/s(n))$ . Start by partitioning  $[n]$  into  $n/s(n)$  blocks of  $s(n)$  elements each. Do  $n/s(n)$  passes over  $L$ , where in the  $i$ th pass, check which numbers in the  $i$ th block do not occur in  $L$ : this can be done in  $s(n)$  bits of space by simply keeping a bit vector. Hence we would say NOE has *time-space product*  $O(n^2)$  for all  $n \leq t(n) \leq n^2$ .

Our first main theorem builds on the aforementioned work of Beame to show that this time-space product is optimal, even when programs have access to a random oracle and can err with high probability.

► **Theorem 1.** *For all  $p \in (0, 1]$ , every random oracle  $n$ -way branching program family of size  $2^{s(n)}$  and height  $t(n)$  computing NOE with success probability  $p$  has  $t(n) \cdot s(n) \geq \Omega(n^2)$  for all sufficiently large  $n$ .*

(Note, we need to formally define what a “random oracle branching program” even means. For now, think of it as a non-uniform version of space-bounded computation with constant-time access to a uniform random  $2^{s(n)}$ -bit string. The preliminaries in Section 2 give full detailed definitions.) Using a standard translation of word-RAMs into branching programs, it follows that every probabilistic word-RAM with a random oracle and wordsize  $\log(n)$  computing NOE in time  $t(n)$  and space  $s(n)$  must have  $t(n) \cdot s(n) \geq \Omega(n^2)$  in order to have any non-zero constant success probability.

### Lower Bound for Sorting and Circuit Evaluation

Informally, we say that a reduction from a problem  $A$  to a problem  $B$  is *succinct* if any particular output bit of the reduction can be computed in  $\text{poly}(\log n)$  time with random access to the input. (Definitions can be found in the Preliminaries.) As a warm-up, in Theorem 24 of the paper, we use a simple succinct reduction to obtain an analogous random oracle lower bound for the problem of sorting  $n$  unordered elements from  $[n]$ , which we call SORT. (A tight lower bound without random oracles was proved by Beame [8].) The sorting lower bound is combined with another simple reduction to obtain an analogous lower bound for evaluating a circuits. Let FCIRCEVAL be the problem: *Given a circuit of size  $n$  with at most  $n$  inputs (all fixed to 0-1 values) and at most  $n$  outputs, determine its output.* The decision version of FCIRCEVAL is well-known to be P-complete.

► **Theorem 2.** *For all finite  $\Sigma$ , there is a  $k > 0$  such that the random oracle  $\Sigma$ -way branching program time-space product of FCIRCEVAL is at least  $\Omega(n^2/\log^k n)$ .*

As in the case of NOE, Theorem 2 implies analogous time-space lower bounds on probabilistic random access machines computing FCIRCEVAL. We note that without the “random oracle” modifier, Theorem 2 is almost certainly folklore; it follows from our simple reduction and the  $\tilde{\Omega}(n^2)$  time-space product lower bound on sorting of Borodin and Cook [13].

<sup>3</sup> As usual, we define  $[n] := \{1, \dots, n\}$ .

### Lower Bound for Printing SAT Assignments

Any practical algorithm for Satisfiability would need to print satisfying assignments when they exist. We give a succinct reduction from Circuit Evaluation to the printing problem for 3SAT, proving a nearly-quadratic time lower bound in the  $n^{o(1)}$  space setting.

► **Theorem 3.** *For all finite  $\Sigma$ , there is a  $k > 0$  such that the random oracle  $\Sigma$ -way branching program time-space product of PRINT-3SAT is at least  $\Omega(n^2/\log^k n)$ .*

### Lower Bound for Computing Truth Tables of CNFs

We show that the trivial algorithm for computing the truth table of a large CNF has optimal time-space product, even for randomized algorithms.

► **Theorem 4.** *For all finite  $\Sigma$ , there is a  $k > 0$  such that the random oracle  $\Sigma$ -way branching program time-space product of TTPRINT for CNF formulas with  $n$  clauses and  $\log(n) + \log \log(n)$  many variables is at least  $\Omega(n^2/\log^k n)$ .*

The proof of Theorem 4 works by giving a succinct reduction from NOE to TTPRINT, constructing a CNF whose truth table encodes (at the bit level) the non-occurring elements of a given list.

### Lower Bound for #2SAT

Finally, we give a succinct reduction from the truth table problem TTPRINT for CNF formulas to *counting* SAT assignments to a given 2CNF, implying an analogous lower bound for #2SAT. In particular, we encode the output 2CNF in such a way that the bits of its number of satisfying assignments encode the *value* of the original CNF on various inputs.

► **Theorem 5.** *For all finite  $\Sigma$ , there is some  $k > 0$  such that the random oracle  $\Sigma$ -way branching program time-space product of #2SAT is at least  $\Omega(n^2/\log^k(n))$ .*

### Lower Bounds Beyond?

Finally, we note that the random oracle model we consider may not be the most general possible one. We could also consider oracles where the queries are on write-only storage that does not count towards the space bound. Although such oracles are sometimes used in space-bounded complexity ([20]), it is hard for us to see how such queries could help in the *random* oracle setting. In the paper’s conclusion (Section 5) we outline how to coherently define this sort of oracle access in branching programs, and conjecture that our lower bounds also hold in this “extended oracle” model.

## 1.1 Intuition for the NOE Lower Bound

Our proof of the NOE lower bound (Theorem 1) starts from Beame’s lower bound for UNIQUE ELEMENTS [8]. We generalize his proof, and use our generalization to show that any function problem satisfying two basic properties has a non-trivial branching program lower bound, even when the branching program has access to a huge number of random bits.

► **Theorem 6.** *Let  $\{f_n : \Sigma_n^n \rightarrow \Sigma_n^*\}$  be a family of functions,  $\{D_n\}$  be a family of distributions, and let  $g : \mathbb{N} \rightarrow \mathbb{N}$  satisfy the following properties.*

1. [***f* typically has “long” outputs**]. For all  $\varepsilon > 0$ , there is an  $n_0 \geq 0$  such that for all  $n > n_0$ , there is a  $\delta > 0$  such that

$$\Pr_{x \in D_n} [|f_n(x)| > \delta g(n)] > 1 - \varepsilon.$$

2. [**Short random-oracle branching programs have low probability of printing long substrings of *f***]. Let  $U_n$  be the uniform distribution over  $\Sigma_n$  and let  $N \leq 2^{s(n)}$  be an integer. There is an  $\varepsilon > 0$  such that for all  $\Sigma_n$ -way branching programs  $P$  of height at most  $n/4$ ,

$$\Pr_{(x,r) \sim D_n \times U_n^N} [(P(xr) \text{ is a substring of } f_n(x)) \wedge (|P(xr)| \geq m)] < e^{-\varepsilon m}.$$

Then, for all  $n > n_0$  and  $p \in (0, 1]$ , and for every random oracle  $\Sigma_n$ -way branching program of size  $2^{s(n)}$  and height  $t(n)$  computing  $f_n$  with success probability at least  $p$  on inputs drawn from  $D_n$ , it must be that  $t(n) \cdot s(n) \geq \Omega(n g(n))$ .

The intuition behind Theorem 6 is that, if a function  $f$  on input  $x$  requires a long output (Property 1), then some (possibly many) subprograms of an efficient branching program  $P$  computing  $f$  will need to produce somewhat-long output. But by Property 2, this is “hard” for all short subprograms even with a random oracle: they have low probability of correctly answering a large fraction of  $f$ .

Beame’s original lower bound for UNIQUE ELEMENTS [8] follows a similar high-level pattern (as we review in Theorem 21). One of our insights is that the NON-OCCURRING ELEMENTS problem satisfies stronger versions of the properties used in his proof for UNIQUE ELEMENTS; these stronger properties give us extra room to play with the computational model in our lower bound against NON-OCCURRING ELEMENTS. Another insight is that the conditioning on uniform random input in his argument can be modified to accommodate very long auxiliary random inputs. Together, this allows us to extend the lower bounds to models equipped with random oracles.

### Why Random Oracles?

Let us give one comment on the model itself. One may wonder why it is even necessary to add random oracles to branching programs (BPs), which are a *non-uniform* model of computation. Can’t we use Adleman’s argument [5] to show that the randomness can be hard-coded in the non-uniform model, similarly to how  $\text{BPP} \subset \text{P/poly}$ ?

Indeed, a random-oracle BP can always be simulated by a deterministic BP; however, the running time of the deterministic BP increases by a multiplicative factor of  $\Omega(n)$  (hence, a time lower bound of  $\Theta(n^2)$  on a deterministic BP says nothing *a priori* about randomized BPs). Given a randomized BP for a decision problem with constant success probability  $1/2 < p < 1$ , it is derandomized by taking  $\Omega(n)$  copies of the BP (with independent random bits filled in each copy), and computing the majority value of the BP outputs. The  $\Omega(n)$  copies are needed because one needs to guarantee correctness over all  $2^n$  possible inputs: this increases the running time by an  $\Omega(n)$  multiplicative factor. In our case, the situation is even more complicated because we are concerned with function problems. The upshot is that  $n^2$  lower bounds on random-oracle BPs give strictly more information than a typical randomized model with one-way access to random bits, or a deterministic model.

Another randomized branching program model considered in prior work (e.g., [9]) is to define a randomized time- $T$  space- $S$  branching program to be a *distribution*  $\mathcal{D}$  of deterministic time- $T$  and space- $S$  BPs. Such a model is said to compute a function  $f$  with error  $\varepsilon$  if, on



every input  $x$ , a randomly drawn BP from  $\mathcal{D}$  outputs  $f(x)$  with probability at least  $1 - \varepsilon$ . This model looks even more general than our random oracle model, since no resources are spent accessing randomness (the randomness is “drawn” prior to any computation, then fixed for free). An anonymous reviewer for ITCS observed that our time-space lower bound for NON-OCCURRING ELEMENTS also holds in this BP model. The idea is to apply Yao’s principle [29] which reduces worst-case lower bounds for distributions of BPs to finding a hard distribution of inputs for deterministic BPs, and to verify that our proofs essentially show that the uniform distribution is hard for deterministic BPs.

## 2 Preliminaries

We assume basic familiarity with computational complexity [5]. Here we recall (and introduce) various computational models and notations needed in the paper.

All of our lower bounds are on the product of time and space for various problems. For clarity, we define the time-space product as follows.

► **Definition 7.** Let  $f$  be a function. We say that *the time-space product of  $f$  is at least  $b(n)$*  if for every algorithm running in  $t(n)$  time and  $s(n)$  space for  $f$ , it must be the case that  $t(n) \cdot s(n) \geq \Omega(b(n))$ .

### Random Access Machines With Oracles

For this work, we consider random access machines (RAMs) with access to an oracle and a write-only output tape. More precisely, our RAMs can only append new characters to the end of their output tape.

► **Definition 8.** Let  $O$  be a language over a finite alphabet. An **oracle RAM**  $M^O$  is a random access machine with an additional *oracle tape*. During a time step, in addition to any normal RAM operations, an oracle RAM can read or write a character to the oracle tape, can move the oracle tape head left or right, or can query the oracle with the contents of the tape to obtain a uniform random bit. Additionally,  $M^O$  has a write-only output tape to which it can append a character in any step.

### R-Way Branching Programs With Oracles and Output

Our lower bounds for NON-OCCURRING ELEMENTS (Theorem 1) will be against a generalization of branching programs with a (large) alphabet of size  $R$ , often called  $R$ -way branching programs [13]. We recall the definition here.

► **Definition 9.** Let  $R \geq 2$  be an integer. An  **$R$ -way branching program** with  $n$  inputs  $x_1, \dots, x_n$  is a directed acyclic graph with one source node in which every non-sink node has out-degree  $R$ . Every non-sink node is labeled with an index  $i \in [n]$  corresponding to an input variable, and each edge is labeled with an element of  $[R]$  such that no edge  $(u, v)$  and  $(u, w)$  where  $v \neq w$  share a label. Additionally, each vertex  $v$  is labeled with an instruction to print a value  $p(v)$  (which may be empty). The **height** of an  $R$ -way branching program  $P$  is the length of the longest path in its graph, and the **size** is the number of vertices. The **computation path of  $P$  on input  $x$**  is the unique path  $\pi = (v_0, v_1), (v_1, v_2), \dots, (v_{k-1}, v_k)$  such that  $v_0$  is the source node,  $v_k$  is a sink node, and for all  $i \in \{0, \dots, k-1\}$ , if vertex  $v_i$  is labeled  $j$ , then the label on  $(v_i, v_{i+1})$  is the value of  $x_j$ . The **output of  $P$  on input  $x$** , denoted as  $P(x)$ , is the concatenation of the values printed by the vertices along the computation path of  $P$  on input  $x$ , i.e.,  $p(v_0)p(v_1) \cdots p(v_k)$ .



Let  $\Sigma$  be a finite alphabet. For notational convenience, we call a  $P$  a  $\Sigma$ -way branching program if  $P$  is a  $|\Sigma|$ -way branching program with edges labeled with elements of  $\Sigma$  instead of  $[|\Sigma|]$ .

► **Definition 10.** A function  $f : \Sigma^* \rightarrow \Sigma^*$  has a  $\Sigma$ -way branching programs of height  $t(n)$  and size  $S(n)$  if for all  $n \geq 0$ , there is a  $\Sigma$ -way branching program  $P_n$  of height  $t(n)$  and size  $S(n)$  such that for all  $x \in \Sigma^n$ ,  $P_n(x) = f(x)$ .

We study a natural generalization of  $R$ -way branching programs with oracles whose queries count towards the space bound. This is a natural set-up for the random oracle setting, where random oracles model truly random hash functions. Let  $O : \Sigma^* \rightarrow \{0, 1\}$  in the following.

► **Definition 11.** An  $O$ -oracle  $R$ -way branching program with  $n$  inputs is an  $R$ -way branching program with the following additional properties. Each vertex is labeled with either a *variable index*  $i \in [n]$  or with a string  $q_j \in \{0, 1\}^*$  (where  $j$  is an integer ranging from 1 to the number of vertices in the program); we call the latter  $Q$ -vertices (for “Query”). All  $Q$ -vertices have two outgoing arcs, labeled with the two possible query answers *yes* or *no* (i.e., 1 or 0). Computation on an  $O$ -oracle branching program is defined analogously to usual branching programs, with the following extra rule for  $Q$ -vertices. Each time a  $Q$ -vertex  $v$  is reached during a computation, the outgoing *yes* (i.e., 1) edge of  $v$  is taken in the computation path if and only if the label  $q_j$  of  $v$  satisfies  $O(q_j) = 1$ .

► **Remark.** Note that in the above definition, the oracle queries could be strings of *arbitrary* length, but in a size- $S$  branching program we are only allowed  $S$  possible distinct oracle queries over all possible inputs. So the above definition is more general than the condition that “oracle queries count towards the space bound”. In the random oracle setting, this distinction will make little difference; we might as well think of the query strings as being of length about  $\log_2(S)$ .

It is natural to augment branching programs with oracles. Barrington and McKenzie [6], motivated by an approach to proving  $\text{NC}^1 \neq \text{P}$ , defined an oracle branching program model in terms of finite automata, where instead of branching on individual input bits, the program can branch on (in principle) all possible  $n$ -bit inputs, corresponding to oracle queries on the input. The usual branching program model is captured in their model by a branching program with an oracle for the predicate  $\text{BIT}(x, i)$  which returns the  $i$ th bit of the input  $x$ . They proved exponential size lower bounds for branching programs with certain weak oracles. Our oracle model is designed to give a natural correspondence between  $O$ -oracle  $R$ -way branching programs and word RAMs with wordsize  $\log(R)$  and oracle access to  $O$ .

It is helpful to think of oracle branching programs as simply branching programs *which receive the oracle as part of their input*. The following proposition formalizes this:

► **Proposition 2.1.** Let  $P^O$  be an  $O$ -oracle  $\Sigma$ -way branching program with  $n$  inputs, height  $T$ , and size  $S$ , and let  $q_1, \dots, q_m$  be the set of all possible oracle queries appearing at  $Q$ -vertices of  $P^O$ . There is a  $\Sigma$ -way branching program  $P'$  of height  $T$  and size  $S$  such that for all  $x \in \Sigma^n$ ,  $P^O(x) = P'(xy)$ , where  $y = O(q_1), \dots, O(q_m)$ .

**Proof.** The branching program  $P'$  is simply a relabeling of  $P$  in which every  $Q$ -vertex with label  $q_j$  is instead labeled by the variable index  $n + j$ . ◀

### Random Oracles

We will ultimately study  $R$ -way branching programs with random oracles. All of our random oracles will have the form  $O : \Sigma^* \rightarrow \Sigma$  for a finite alphabet  $\Sigma$ . Thus we define random oracles as a random elements of the set  $(\Sigma^* \rightarrow \Sigma)$ .

► **Definition 12.**  $\mathcal{D}_\Sigma$  is the uniform distribution over functions in  $(\Sigma^* \rightarrow \Sigma)$ . That is, drawing some  $f \sim \mathcal{D}_\Sigma$  is equivalent to, for each  $x \in \Sigma^*$ , choosing an element of  $\Sigma$  uniformly at random as  $f(x)$ .

We define computation with a random oracle branching program as follows.

► **Definition 13.** A function  $f$  has a **random oracle branching program family of height  $t(n)$  and size  $2^{s(n)}$  with success probability  $p \in (0, 1]$**  if there is a family of oracle branching programs  $\{P_n^O\}$  such that the height of each  $P_n^O$  is at most  $t(n)$ , the size of each  $P_n^O$  is at most  $2^{s(n)}$ , and for all inputs  $x$ ,

$$\Pr_{O \sim \mathcal{D}_\Sigma} \left[ P_{|x|}^O(x) = f(x) \right] \geq p.$$

Borodin and Cook show that  $R$ -way branching programs of height  $O(t(n))$  and size  $2^{O(s(n))}$  can simulate time- $t(n)$  space- $s(n)$  RAMs of wordsize  $\lceil \log_2 R \rceil$  [13]. We observe that their proof relativizes to allow oracle  $R$ -way branching programs to simulate oracle RAMs with wordsize  $\lceil \log_2 R \rceil$ .

► **Lemma 14.** *For every language  $O$  and oracle RAM  $M^O$  running in time  $t(n) \leq 2^{O(s(n))}$  and space  $s(n) \geq \log(n)$  with wordsize  $\log(\Sigma)$ , there is an oracle  $\Sigma$ -way branching program of height  $O(t(n))$  and size  $2^{O(s(n))}$  such that  $M^O(x) = P^O(x)$  for all inputs  $x$ .*

**Proof.** Without loss of generality, assume that at any step,  $M^O$  either accesses its input of length  $n$  by writing an index  $j \in [n]$  to an “access register”, or  $M^O$  queries  $O$  by writing the character  $Q$  to the access register. Let the configuration of machine  $M$  contain the description of the random access memory of  $M$ , the state of  $M$ , and the character currently being output, if any. For integer  $n \geq 1$ , define a graph  $G_n = (V_n, E_n)$  as follows. Let

$$V_n = \{(C, i, j) \mid C \text{ is a space-}s(n) \text{ configuration of } M, i \in \{0, 1, \dots, t(n)\}, j \in [n] \cup \{Q\}\}.$$

Our corresponding branching program, has each vertex  $(C, i, j)$  with  $j \in [n]$  labeled by  $j$ , and those  $(C, i, j)$  with  $j = Q$  are  $Q$ -vertices, which we label with a string  $q_j$  representing the content of the oracle tape of  $M^O$  in configuration  $C$ .

For  $j \in [n]$ , we put the edge  $((C_1, i, j), (C_2, i + 1, j')) \in E_n$  and label it with a string  $p$  if and only if  $M^O$  in configuration  $C_1$  would transition to configuration  $C_2$  given that in configuration  $C_1$ ,  $j$  is put in the access register,  $x_j = p$ , and  $j'$  is put in the access register in  $C_2$ . For configurations making an oracle query, put  $((C_1, i, Q), (C_2, i + 1, j')) \in E_n$  and label it with  $b \in \{0, 1\}$  if and only if  $M^O$  in configuration  $C_1$  would transition to configuration  $C_2$  given that in configuration  $C_1$ ,  $Q$  is written on the access register in  $C_1$ , the oracle query  $O$  returns  $b$ , and  $j'$  is written on access register in  $C_2$ .

Observe that  $|V_n| \leq 2^{O(s(n))} \cdot t(n) \cdot n \leq 2^{O(s(n))}$ , assuming the size of the tape alphabet and the number of states in the state machine of  $M^O$  are constants. Further observe that all paths in  $G_n$  have length at most  $t(n)$ , since every step in the path must be from some  $(C, i, j, k)$  to some  $(C', i + 1, j', k')$  and  $i, i + 1 \in [t(n)]$ . Finally, the oracle branching program  $P_n^O$  is defined to be the  $\Sigma$ -way branching program which is the induced subgraph of  $G_n$  containing all vertices reachable from  $v_0 = (C_0, 0, j_0)$  (with the labels prescribed above), where  $C_0$  is the initial configuration of  $M^O$ , and  $j_0$  is the index of the input read in the initial configuration. We see that by construction, for all  $x \in \Sigma^n$ ,  $P_n^O(x) = M^O(x)$ . ◀

Lemma 14 immediately implies that  $R$ -way branching program lower bounds provide analogous RAM lower bounds:

► **Proposition 2.2.** *Let  $f : \Sigma^* \rightarrow \Sigma^*$ . Suppose there is no  $R$ -way branching program family of height  $O(t(n))$  and size  $2^{O(s(n))}$  computing  $f$ . Then there is no RAM of wordsize  $\log(R)$  running in time  $t(n)$  with space  $s(n)$  computing  $f$ .*

*Furthermore, if there is no random oracle  $R$ -way branching program family of height  $O(t(n))$  and size  $2^{O(s(n))}$  computing  $f$  with success probability  $p$ , then there is no oracle RAM  $M^O$  with wordsize  $\log(R)$  such that for all  $x \in \Sigma^*$ ,  $\Pr_{O \sim \mathcal{D}_\Sigma}[M^O(x) = f(x)] \geq p$ .*

### Succinct Reductions

After our lower bound for NOE has been proved, we can apply very efficient reductions from NOE to extend the lower bound to other problems. For this purpose, we need some notation. In what follows, let  $f$ ,  $g$ , and  $\pi$  be functions from  $\Sigma^*$  to  $\Sigma^*$ .

► **Definition 15.** We say  $A$  has a  $t(n)$ -time reduction with blowup  $b(n)$  to  $B$  if there is a many-one reduction  $f$  reducing  $A$  to  $B$  such that  $|f(x)| \leq b(|x|)$ , and any character in the string  $f(x)$  can be computed by an algorithm running in time  $t(|x|)$ , given the index  $i = 1, \dots, |f(x)|$  of the character and random access to  $x$ .

Polylog-time reductions with quasi-linear blowup essentially preserve lower bounds for our branching program model, up to polylog factors. As the proof is relatively straightforward and our space is limited, the proof is omitted (but appears in the full version).

► **Lemma 16.** *Let  $\pi$  be an  $O(\log^k(n))$ -time reduction with blowup  $b(n)$  from  $f$  to  $g$ , and suppose  $g$  has an oracle branching program family  $\{P_n^O\}$  of height  $t(n)$  and size  $2^{s(n)}$  with success probability  $p > 0$ . Then  $f$  has an oracle branching program family  $\{Q_n^O\}$  of height  $t(b(n)) \log^k(n)$  and size  $2^{O(s(b(n)) + \log^k(n))}$  with success probability at least  $p$ .*

► **Definition 17.** The **random oracle  $\Sigma$ -way branching program time-space product of  $f$  is at least  $b(n)$**  if for every constant  $p \in (0, 1]$  and every random oracle branching program family of height  $t(n)$  time and size  $2^{s(n)}$  computing  $f$  with success probability  $p$ , it must be that  $t(n) \cdot s(n) \geq \Omega(b(n))$ .

We also note (proof omitted in this version) that lower bounds on time-space products are roughly preserved by polylog-time reductions of quasi-linear blowup.

► **Lemma 18.** *Let  $\pi$  be an  $O(\log^j(n))$ -time reduction with blowup  $O(n \log^j(n))$  from  $f$  to  $g$ . Suppose there are  $k$  and  $d$  such that the random oracle  $\Sigma$ -way branching program time-space product of  $f$  is at least  $\Omega(n^d) / \log^k(n)$ . Then there is some  $k'$  such that the random oracle  $\Sigma$ -way branching program time-space product of  $g$  is at least  $\Omega(n^d) / \log^{k'}(n)$ .*

For our #2SAT lower bound, we inspect a reduction of Hoffmeister, and note that it is succinct.

► **Theorem 19** ([19]). *There is a  $\log(n)$ -time reduction from #3-SAT to #2-SAT with blowup  $\tilde{O}(n)$ .*

Finally, our lower bound for FCIRCEVAL exploits the fact that there are highly efficient circuits for sorting  $n$  items from  $[n]$ .

► **Theorem 20** ([7, 26, 15]). *There is a  $k$  such that SORT has  $O(\log^k(n))$  time uniform circuits of size  $O(n \log^k(n))$ .*

### Other Related Work

Besides the work cited earlier, there is an extensive literature on proving  $\Omega(\tilde{n}^2)$  time-space product lower bounds for computing functions in generic word-RAM-like models (usually by proving a  $R$ -way branching program lower bound). The functions include matrix multiplication and the discrete Fourier transform [30], generalized string matching [1], bit-vector convolution and integer multiplication [2], universal hashing from  $n$  bits to  $O(n)$  bits [21], and computing various functions over sliding windows [9].

## 3 Lower Bound for NOE and Sorting

In this section, we abstract out key properties of the UNIQUE ELEMENTS problem that were used in Beame’s proof of an  $\Omega(n^2)$  time-space product lower bound for UNIQUE ELEMENTS against  $R$ -way branching programs [8]. This abstraction is useful in two ways. First, it allows us to easily prove lower bounds for other problems, by simply verifying that the key properties hold. Second, this level of abstraction helps us identify stronger generalizations of the lower bounds: average-case lower bounds against  $R$ -way branching programs and RAMs with random oracles.

### 3.1 Beame’s Method (Without Random Oracles)

To give intuition for our lower bound theorem for branching programs with random oracles (Theorem 6), we begin with a similar but weaker theorem, which is an abstraction of the technique used by Beame [8].

► **Theorem 21.** *Let  $\{f_n : \Sigma_n^n \rightarrow \Sigma_n^*\}$  be a family of functions,  $\{D_n\}$  be a family of distributions, and  $g : \mathbb{N} \rightarrow \mathbb{N}$  with the following properties.*

1. *[ $f$  typically has “long” outputs]. There is an  $\varepsilon > 0$  and  $\delta > 0$  such that*

$$\Pr_x [ |f(x)| > \delta g(n) ] > \varepsilon.$$

2. *[Short branching programs have low probability of printing long substrings of  $f$ ]. There is an  $\varepsilon > 0$  such that, for all  $\Sigma_n$ -way branching programs  $P$  of height at most  $n/4$ , and for all  $m \geq 1$ ,*

$$\Pr_{x \sim D} [ P(x) \text{ is a substring of } f(x) \wedge |P(x)| \geq m ] < e^{-\varepsilon m}.$$

*Then for  $n > n_0$ , every  $\Sigma_n$ -way branching program of size  $2^{s(n)}$  and height  $t(n)$  for  $f_n$  has  $s(n)t(n) \geq \Omega(ng(n))$ .*

Theorem 21 is motivated by the observation that if a function  $f$  on input  $x$  requires a long output (Property 1), then some (possibly many) subprograms of a branching program  $P$  computing  $f$  need to output a large fraction of  $f$ . But by Property 2, this is “hard” for all short subprograms: they have low probability of correctly answering a large fraction of  $f$ .

**Proof of Theorem 21.** Let  $f, D, g(n), \varepsilon, \delta$  be given as above. We prove the lower bound by demonstrating that any space- $S$  branching program  $P$  of sufficiently low height  $T$  has a nonzero probability of error on an input  $x$  drawn from  $D$ . To do this, we lower bound the probability of error by

$$\Pr_{x \sim D} [ |f(x)| > \delta g(n) ] - \Pr_{x \sim D} [ (|f(x)| > \delta g(n)) \wedge (P(x) = \text{NOE}(x)) ].$$

By Property 1 of the hypothesis, the first term is lower bounded by some constant  $\varepsilon > 0$ . The second term is at most  $\Pr_{x \sim D}[|P(x)| > \delta g(n)]$ , giving us an error probability of at least

$$\varepsilon - \Pr_{x \sim D}[|P(x)| > \delta g(n)].$$

We now upper-bound  $\Pr_{x \sim D}[|P(x)| > \delta g(n)]$ .

Without loss of generality, let  $P$  be layered, having size  $2^S$  in each layer, and height  $T$ . Partition  $P$  into  $4T/n$  layers of height  $n/4$ . By the pigeonhole principle, some layer must output at least  $ng(n)/4T$  elements of the output. There are at most  $2^S$  such subprograms in that layer, and the probability that  $P$  outputs at least  $\delta g(n)$  elements correctly is upper bounded by the probability that some layer outputs  $m := \delta ng(n)/4T$  elements correctly. As there are at most  $2^S$  such subprograms, by a union bound over property 2 of the hypothesis, the probability that  $P$  outputs all elements correctly is upper bounded by

$$2^S e^{-\varepsilon' \delta ng(n)/4T}$$

for some  $\varepsilon' > 0$ . This implies that the error probability of  $P$  on an input  $x$  drawn from  $D$  is at least  $\varepsilon - 2^S e^{-\varepsilon' \delta ng(n)/4T}$ .

Finally, we observe that if  $ST \leq \alpha \cdot ng(n)$  for sufficiently small  $\alpha > 0$ , the term  $2^S e^{-\varepsilon' \delta ng(n)/4T}$  becomes less than  $\varepsilon$ . Thus there is some input length  $n$  such that the probability of error for  $P$  on an input  $x$  drawn from  $D$  is nonzero. This implies that  $ST \geq \Omega(ng(n))$ . ◀

Beame's lower bound against UNIQUE ELEMENTS follows from showing that UNIQUE ELEMENTS has the properties required by Theorem 21. In particular, on random inputs UNIQUE ELEMENTS has long outputs with high probability, and it is difficult to guess even a small number of elements in the output of a random input, without seeing most of the input. Instead of reproving the UNIQUE ELEMENTS lower bound, we give a lower bound against the problem NON-OCCURRING ELEMENTS (NOE) defined in the introduction, as it admits a slightly easier analysis.

First let us verify Property 1 of Theorem 21, for the uniform distribution  $U_n^n$  and  $g(n) = n$ .

► **Proposition 3.1** (Property 1 holds for NOE). *For all  $\varepsilon > 0$ , there is a  $\delta > 0$  such that for sufficiently large  $n$ ,  $\Pr_{x \in U_n^n}[|NOE(x)| > \delta n] > 1 - \varepsilon$ .*

**Proof.** The desired bound reduces to a weak version of a well-known Balls-and-Bins bound (see [16, p.75] for a reference). In particular, when selecting  $m$  integers from  $[n]$  uniformly at random, the number  $Z$  of integers in  $[n]$  not selected (the non-occurring elements) is tightly concentrated around its mean:

$$\Pr[|Z - E[Z]| > t] \leq 2 \exp(-2t^2/m).$$

For our problem, we have  $n = m$  and  $E[Z] = n/e$ . Let  $c > 0$  be a parameter, and let  $t = cn$ . Therefore we have

$$\Pr[|Z - n/e| > cn] \leq 2 \exp(-2c^2n).$$

Complementing and rearranging variables, the inequality becomes  $\Pr[Z \geq n/e - cn] \geq 1 - 2 \exp(-2c^2n)$ . Finally, for any  $\varepsilon > 0$ , we can pick  $c \in (0, 1/e)$  such that  $\varepsilon > 2 \exp(-2c^2n)$  for sufficiently large  $n$ . Letting  $\delta \in (0, 1/e - c)$ , we have  $\Pr[Z > \delta n] > 1 - \varepsilon$ . ◀

Note that the bound of Proposition 3.1 is stronger than that required by Theorem 21. This will be useful for the extension to random oracles later (Theorem 6).

Next, we verify that for NON-OCCURRING ELEMENTS, Property 2 of Theorem 21 holds as well. In fact, we prove a stronger statement that allows for extra side randomness in the input (and therefore random oracle branching programs). This randomness can be ignored in the application of Theorem 21.

► **Proposition 3.2** (Property 2 holds for NOE). *For all integers  $n, k, N \geq 0$  and all branching programs  $P$  of height at most  $n/4$  computing a function with  $m$  outputs,*

$$\Pr_{x \sim U_n^N, r \sim U_k^N} [P(xr) \text{ outputs } m \text{ non-occurring elements of } x] < e^{-3m/4}.$$

**Proof.** The desired probability equals

$$\sum_{\text{paths } \pi \text{ in } P} \Pr_{x,r} [P(xr) \text{ follows } \pi] \cdot \Pr_{x,r} [P(xr) \text{ has } m \text{ NOEs of } x \mid P(xr) \text{ follows } \pi].$$

We show that for all such  $\pi$ ,

$$\Pr_{x,r} [P(xr) \text{ has } m \text{ non-occurring elements of } x \mid P(xr) \text{ follows } \pi] < e^{-3m/4}.$$

From this, it will follow that our desired probability is less than  $e^{-3m/4}$ .

For a path  $\pi$ , let  $q$  be the number of distinct variable queries to  $x$  and let  $q'$  be the number of distinct queries made to  $r$ . Notice that  $q + q' \leq n/4$ . To bound the probability that  $P(xr)$  has at least  $m$  non-occurring elements of  $x$ , given that  $P(xr)$  follows  $\pi$ , we simply count the relevant numerator and denominator.

The total number of  $xr$  that follow  $\pi$  is  $n^{n-q} \cdot k^{N-q'}$ : there are  $n - q$  unqueried inputs of  $x$ , and  $N - q'$  unqueried inputs of  $r$ . The total number of  $xr$  that follow  $\pi$ , and for which the  $m$  outputs of  $\pi$  are non-occurring elements of  $x$ , is at most  $(n - m)^{n-q} \cdot k^{N-q'}$ : since the  $m$  outputs are supposed to be non-occurring in  $x$ , none of the remaining  $n - q$  unqueried variables can take on any of the  $m$  outputs. By simple manipulation, we have

$$\frac{(n - m)^{n-q} k^{N-q'}}{n^{n-q} k^{N-q'}} = (1 - m/n)^{n-q} \leq (1 - m/n)^{n-n/4} \leq (1 - m/n)^{3n/4} \leq e^{-3m/4}.$$

This completes the proof. ◀

## 3.2 Lower Bounds With Random Oracles

We are now ready to present our main lower bound theorem against branching programs with random oracles.

► **Reminder of Theorem 6.** *Let  $\{f_n : \Sigma_n^n \rightarrow \Sigma_n^*\}$  be a family of functions,  $\{D_n\}$  be a family of distributions, and let  $g : \mathbb{N} \rightarrow \mathbb{N}$  satisfy the following properties.*

1. [***f* typically has “long” outputs**]. *For all  $\varepsilon > 0$ , there is an  $n_0 \geq 0$  such that for all  $n > n_0$ , there is a  $\delta > 0$  such that*

$$\Pr_{x \in D_n} [|f_n(x)| > \delta g(n)] > 1 - \varepsilon.$$

2. [***Short random-oracle branching programs have low probability of printing long substrings of f***]. *Let  $U_n$  be the uniform distribution over  $\Sigma_n$  and let  $N \leq 2^{s(n)}$  be an integer. There is an  $\varepsilon > 0$  such that for all  $\Sigma_n$ -way branching programs  $P$  of height at most  $n/4$ ,*

$$\Pr_{(x,r) \sim D_n \times U_n^N} [(P(xr) \text{ is a substring of } f_n(x)) \wedge (|P(xr)| \geq m)] < e^{-\varepsilon m}.$$



Then, for all  $n > n_0$  and  $p \in (0, 1]$ , and for every random oracle  $\Sigma_n$ -way branching program of size  $2^{s(n)}$  and height  $t(n)$  computing  $f_n$  with success probability at least  $p$  on inputs drawn from  $D_n$ , it must be that  $t(n) \cdot s(n) \geq \Omega(ng(n))$ .

Theorem 6 prescribes a general scheme for proving time-space product lower bounds against random oracle  $R$ -way branching programs, and thus against word RAMs with random oracles as well.

**Proof of Theorem 6.** The proof is similar to Theorem 21; we focus on highlighting what is different. First, we note that by Proposition 2.1, it is sufficient to demonstrate that for all  $n > n_0$  and  $p \in (0, 1]$ , and for every  $\Sigma_n$ -way branching program of size  $2^{s(n)}$  and height  $t(n)$  where  $\Pr_{x \sim D_n, r \sim U_n^N}[P(xr) = f_n(x)] > p$ , it must be that  $t(n) \cdot s(n) \geq \Omega(ng(n))$ .

As in Theorem 21, we establish the lower bound by demonstrating that every  $n/4$ -height branching program  $P$  has a large probability of error on an input  $x \sim D_n$ . In what follows, assume  $P$  successfully computes  $f_n(x)$  on inputs  $(x, r)$  drawn from  $D_n \times U_n^N$  with probability at least  $p$ . By property 1 of the hypothesis, letting  $\varepsilon := p/2$ , there is a  $\delta$  such that  $\Pr_{x \in D, r \sim U_n^N}[|f(x)| > \delta g(n)] > 1 - p/2$ .

Analogously as in the proof of Theorem 21, we lower bound the probability of error of  $P$  on  $D_n$  by

$$\Pr_{x \sim D}[|f(x)| > \delta g(n)] - \Pr_{x \sim D, r \sim U_n^N}[|f(x)| > \delta g(n) \wedge P(xr) = f(x)];$$

note that this probability is at least  $(1 - p/2) - \Pr_{x \sim D, r \sim U_n^N}[|P(xr)| > \delta g(n)]$ .

Applying a union bound over all  $2^{s(n)}$  subprograms of  $P$  as before, but substituting property 2 from the hypothesis, we determine by an analogous argument as Theorem 21 that  $\Pr_{x \sim D, r \sim U_n^N}[|P(xr)| > \delta g(n)]$  is at most  $2^{s(n)} e^{-\varepsilon \delta n g(n)/4t(n)}$ . Therefore the error probability of  $P$  on  $D_n$  is at least

$$1 - p/2 - 2^S e^{-\frac{\varepsilon \delta n g(n)}{4t(n)}}.$$

Finally, if we assume  $s(n)t(n) \leq \alpha n \cdot g(n)$  for all  $\alpha > 0$ , we can tune  $2^{s(n)} e^{-\varepsilon \delta n g(n)/(4t(n))}$  to be arbitrarily small: it is at most  $2^{s(n)} e^{-\varepsilon \delta s(n)/(4\alpha)}$ . Thus we can make the error probability for  $P$  on an input  $x$  drawn from  $D_n$  to be at least  $1 - 2p/3$ , implying that the success probability is at most  $2p/3 < p$ . This is a contradiction, so there is some  $\alpha > 0$  (depending on  $\delta, \varepsilon$ , and  $p$ ) such that  $s(n)t(n) \leq \alpha n \cdot g(n)$ . ◀

► **Remark.** If a function does not satisfy Property 2 of Theorem 6 but satisfies a slightly weaker property, namely that there is an  $\varepsilon > 0$  such that for all  $|\Sigma_n|$ -way branching programs  $P$  of height at most  $n/4$ ,

$$\Pr_{x \sim D_n}[P(x) \text{ is a substring of } f(x) \wedge P(x) \geq m] < e^{-\varepsilon m},$$

we can still obtain an average case lower bound against  $f$  for inputs drawn from  $D_n$ , but not necessarily a lower bound against random-oracle branching programs. We omit an exposition of this result, because we have not yet found applications of it.

We conclude this subsection with the lower bound for NOE with a random oracle.

► **Reminder of Theorem 1.** For every  $p \in (0, 1]$ , every random oracle  $n$ -way branching program family of size  $2^{s(n)}$  and height  $t(n)$  computing NOE with success probability  $p$  must have  $t(n) \cdot s(n) \geq \Omega(n^2)$  for all sufficiently large  $n$ .

**Proof.** Applying Proposition 3.1 and 3.2 and set  $D_n := U_n^n$  and  $\varepsilon := 3/4$ . Then both properties 1 and 2 of Theorem 6 hold for NON-OCCURRING ELEMENTS. ◀



### 3.3 Sort and Random Oracles

By reducing from NON-OCCURRING ELEMENTS to SORT, it follows that random oracle  $n$ -way branching programs still require a time-space product of  $\Omega(n^2)$  to sort a list  $L \in [n]^n$ . (The proof is omitted due to space restrictions.)

► **Theorem 22.** *For any constant  $c \in (0, 1]$ , let  $\{P_n^O\}$  be an oracle  $n$ -way branching program family of size  $2^{s(n)}$  and height  $t(n)$  such that for all  $n, x \in [n]^n$ ,  $\Pr_{O \sim \mathcal{D}_{[n]}}[P_n^O(x) = \text{SORT}(x)] \geq c$ . Then,  $ST = \Omega(n^2)$ .*

To prove our lower bounds for other problems in the following section, we require a somewhat stronger result: a nearly-quadratic time-space product lower bound for computing the non-occurring elements of a list of  $n$  items from  $[n]$ , as well as sorting  $n$  items from  $[n]$  for lists encoded over a *finite* alphabet  $\Sigma$ . By a reduction, we can prove this using the general lower bounds of Theorem 1 and Theorem 22.

► **Lemma 23.** *Let  $\{f_n : \Sigma_n^n \rightarrow \Sigma_n^*\}$  be a family of functions, and let  $f_\Sigma : \Sigma^* \rightarrow \Sigma^*$  be such that  $f(\langle x \rangle_\Sigma) = \langle f_{|x|}(x) \rangle_\Sigma$ , where  $\langle s \rangle_\Sigma$  is  $s$  encoded by a string over  $\Sigma$ . Suppose  $\{f_n\}$  requires a random oracle  $\Sigma_n$ -way branching program time-space product of  $\Omega(n^d)$ . Then there is some  $k > 0$  such that  $f_\Sigma$  requires a random oracle  $\Sigma$ -way branching program time-space product of  $\Omega(n^d / \log^k(n))$ .*

**Proof.** By contradiction. Suppose for all  $k > 0$ ,  $f_\Sigma$  has random oracle  $\Sigma$ -way branching program family  $\{P_n^O\}$  of height  $t(n)$  and size  $2^{s(n)}$  with success probability  $p > 0$  where  $t(n)s(n) \leq O(n^d / \log^k(n))$ . We show that  $\{f_n\}$  has a random oracle  $n$ -way branching program family  $\{Q_n^O\}$  of height  $t'(n)$  and size  $2^{s'(n)}$  with success probability  $p > 0$  where  $t'(n)s'(n) \leq O(n^d / \log^k(n))$ . To do this, we simply simulate  $P_n^O$  with  $Q_n^O$ . We first consider the input  $l \in [n]^n$  a length  $n \lceil \log_{|\Sigma|}(n) \rceil$  string where each number is represented by some string of characters from  $\Sigma$ . We then modify  $P_{n \lceil \log_{|\Sigma|}(n) \rceil}^O$  as follows. For each vertex  $v$  which queries an input  $i$ , instead query input  $\lfloor i / \log_{|\Sigma|}(n) \rfloor$ . Then, for all edges  $(u, v)$  with label  $\alpha \in \Sigma$  in  $P_{n \lceil \log_{|\Sigma|}(n) \rceil}^O$ , add an edge from  $u$  to  $v$  with label  $\beta \in [n]$  for all  $\beta$  whose representation in base  $\Sigma$  contains  $\alpha$  at position  $i \bmod \lceil \log_{|\Sigma|}(n) \rceil$ . Finally, we need only modify the output behavior of  $P_{n \lceil \log_{|\Sigma|}(n) \rceil}^O$ . It prints the representation of characters  $\sigma \in \Sigma_n$  using characters from  $\Sigma$ . We need only  $O(\log(n))$  extra bits of storage to remember the last  $\lceil \log_{|\Sigma|}(n) \rceil$  characters  $P_{n \lceil \log_{|\Sigma|}(n) \rceil}^O$  would have printed, and upon reaching enough characters, we simply print the corresponding element of  $\Sigma_n$  and remember that we have started a new character. As in the proof of Theorem 22, we can do this by creating  $O(n)$  copies of our modified branching program and transitioning accordingly and letting this be  $Q_n^O$ .

Finally, we see that by construction,  $\{Q_n\}$  computes  $\{f_n\}$  with success probability  $p$ . Further, we see that the height of  $Q_n$  is  $O(t(n \log(n)))$  and the size is  $2^{O(s(n) + \log(n))}$ . By our assumption, we see that  $t(n \log(n))(s(n) + \log(n)) \geq \Omega(n^d)$  and  $t(n \log(n))(s(n) + \log(n)) \leq O((n \log(n))^d / \log^{d+1}(n))$ . This is a contradiction. ◀

From this we can conclude lower bounds for NON-OCCURRING ELEMENTS and SORT for strings over finite alphabets.

► **Lemma 24.** *Let  $\text{NON-OCCURRING ELEMENTS}_\Sigma : \Sigma^* \rightarrow \Sigma^*$  be a function which maps the encoding in  $\Sigma$  of a list  $\langle L \rangle$  (where  $L \in [n]^n$  for some  $n$ ) to  $\langle \text{NON-OCCURRING ELEMENTS}(L) \rangle$ , and let  $\text{SORT}_\Sigma : \Sigma^* \rightarrow \Sigma^*$  be the function which maps the encoding of a list  $\langle L \rangle$  where  $\exists n \in \mathbb{N}(L \in [n]^n)$  to  $\langle \text{SORT}(L) \rangle$ . For all  $\Sigma$ , there is some  $k > 0$  such that, the random oracle  $\Sigma$ -way branching program time-space product for both  $\text{NON-OCCURRING ELEMENTS}_\Sigma$  and  $\text{SORT}_\Sigma$  are both at least  $\Omega(n^2 / \log^k n)$ .*

**Proof.** This follows directly from Theorems 1 and 22 and Lemma 23. ◀

► **Remark.** It will be important later that the branching program lower bounds we have established hold even if we allow the branching program to make small perturbations on the output. For example, all above lower bound proofs still go through, if we permit branching programs to output 0 at any node that does not already output some other number. Thus our lower bounds hold for any BP computing NON-OCCURRING ELEMENTS or NON-OCCURRING ELEMENTS $_{\Sigma}$  which prints the binary representation of a list  $L$ , such that  $L$  contains all non-occurring elements of  $x$ , along with any number of elements which are 0.

## 4 Reductions

We now give a series of reductions from NOE and SORT, showing nearly-quadratic time-space product lower bounds for several natural circuit-analysis problems. Each reduction is very efficient, requiring only  $\text{poly}(\log n)$  time to look up any bit of the output of the reduction, and only creating problem instances of size  $\tilde{O}(n)$  from inputs of size  $n$ .

As a warm-up, we observe a very simple reduction from SORT $_{\{0,1\}}$  (sorting  $n \log(n)$ -bit strings) to FCIRCEVAL. Recall in FCIRCEVAL we are given a circuit of size  $n$  with at most  $n$  inputs (all fixed to 0-1 values) and at most  $n$  outputs, and want to determine its output.

► **Reminder of Theorem 2.** *For all finite  $\Sigma$ , there is some  $k > 0$  such that the random oracle  $\Sigma$ -way branching program time-space product of FCIRCEVAL is at least  $\Omega(n^2 / \log^k n)$ .*

**Proof.** The idea is simple: on an unsorted input, we can succinctly produce a circuit for sorting and ask FCIRCEVAL for its output. We observe there is an  $O(\log n)$ -time reduction with blowup  $O(\log^k n)$  from SORT to FCIRCEVAL (for some  $k$ ). Given an input list  $x$  to be sorted, we produce a circuit  $C$  for SORT with  $x$  hard-coded as the input. Any bit of the circuit description can be computed in  $O(\log n)$  time, as SORT has  $O(\log n)$ -time uniform circuits of size  $\tilde{O}(n)$ , and any bit of  $x$  can trivially be produced in  $O(\log n)$  time. By Lemmas 24 and 18, we conclude a time-space lower bound of the form  $\Omega(n^2 / \log^k n)$  for FCIRCEVAL. ◀

A straightforward corollary of Theorem 2 is a similar lower bound against a seemingly weaker version of 3SAT.

► **Definition 25.** Let PROMISE-PRINTING-UNIQUE-SAT be the problem: given a circuit  $C$  promised to have exactly one satisfying assignment, print the satisfying assignment of  $C$ .

► **Lemma 26.** *For all finite  $\Sigma$ , there is a  $k > 0$  such that the random oracle  $\Sigma$ -way branching program time-space product of PROMISE-PRINTING-UNIQUE-SAT is at least  $\Omega(n^2 / \log^k n)$ .*

**Proof.** We give a reduction from FCIRCEVAL to PROMISE-PRINTING-UNIQUE-SAT. Given an instance  $C$  of FCIRCEVAL with only constant inputs and  $m$  output bits  $y_1, \dots, y_m$ , consider the circuit  $C'(x)$  with  $m$  free input bits  $x_1, \dots, x_m$  and one output bit.  $C'$  can be made such that  $C'(x_1, \dots, x_m) = 1$  if and only if  $x_i = y_i$  by using only as many gates as an needed to construct  $C$  and  $O(m)$  additional gates to check equality of the output bits of  $C$  with the free input bits and then to take the conjunction of these equalities. In total, we see then that  $|C'| = O(|C|)$ . Finally we see this gives us an  $O(\log(n))$ -time reduction with blowup  $O(n)$  from FCIRCEVAL to PROMISE-PRINTING-UNIQUE-SAT, since  $\text{FCIRCEVAL}(C) = \text{PROMISE-PRINTING-UNIQUE-SAT}(C')$ , and each bit of  $C'$  can be computed by simply looking up bits of  $C$  directly or deciding if they are part of the equality check or conjunction of the equality checks added to  $C$ . By Lemma 18, we can conclude that PROMISE-PRINTING-UNIQUE-SAT has a time-space lower bound of  $\Omega(n^2 / \log^k n)$ . ◀

To establish a connection with printing satisfying assignments for 3CNFs, we consider the problem PROMISE-PRINTING-UNIQUE-3SAT, which is just PROMISE-PRINTING-UNIQUE-SAT restricted to 3CNF formulas. By another straightforward reduction, it follows that PROMISE-PRINTING-UNIQUE-3SAT cannot be computed more efficiently than FCIRCEVAL.

► **Lemma 27.** *For all finite  $\Sigma$ , there is a  $k > 0$  such that the random oracle  $\Sigma$ -way branching program time-space product of PROMISE-PRINTING-UNIQUE-3SAT is at least  $\Omega(n^2 / \log^k n)$ .*

**Proof.** We present a reduction from the PROMISE-PRINTING-UNIQUE-SAT function to PROMISE-PRINTING-UNIQUE-3SAT. Consider the standard reduction from CIRCUIT SAT to 3SAT in which a circuit  $C$  with  $n$  variables  $x_1, \dots, x_n$  and  $m$  gates is mapped to a CNF  $\phi$  with  $O(m)$  clauses, variables  $x_1, \dots, x_n$  and  $m$  extra variables. Assume without loss of generality (blowing up only an additional constant factor in size otherwise) that  $C$  is comprised only of NAND gates. The typical reduction from CIRCUIT SAT to 3SAT uses an additional variable  $y_1, \dots, y_m$  for each gate  $g_1, \dots, g_m$ , and for each gate  $g_i = \neg(g_j \wedge g_k)$ ,  $g_i = \neg(g_j \wedge x_k)$ , or  $g_i = \neg(x_j \wedge x_k)$ , we add to  $\phi$  the constraints  $(y_i = \neg(y_j \wedge y_k))$ ,  $(y_i = \neg(y_j \wedge x_k))$ , or  $(y_i = \neg(x_j \wedge x_k))$  respectively, where each requires only  $O(1)$  clauses of width 3.

Notice that printing the first  $n$  bits of a satisfying assignment to  $\phi$  is sufficient for computing a satisfying assignment to  $C$ , and observe that this reduction can be done in  $O(\log^k(n))$  time with blowup  $O(\log^k(n))$  for some  $k$ . Analogously to Lemma 18, we can conclude that PROMISE-PRINTING-UNIQUE-3SAT requires a random oracle  $\Sigma$ -way branching program time-space product of  $\Omega(n^2 / \log^{k'}(n))$  for some  $k' > 0$ . ◀

As a corollary, we can immediately conclude that the harder problem of PRINT-3SAT also has a time-space lower bound of  $\Omega(n^2 / \log^k n)$  for some  $k$ .

► **Reminder of Theorem 3.** *For all finite  $\Sigma$ , there is a  $k > 0$  such that the random oracle  $\Sigma$ -way branching program time-space product of PRINT-3SAT is at least  $\Omega(n^2 / \log^k n)$ .*

**Proof.** Follows from the trivial reduction from PROMISE-PRINTING-UNIQUE-3SAT to PRINT-3SAT and Lemma 18. ◀

Next, we show by a reduction directly from NON-OCCURRING ELEMENTS<sub>{0,1}</sub> that printing the truth tables of CNF formulas with a small number of variables admits a time-space lower bound similar to those above.

► **Reminder of Theorem 4.** *For all finite  $\Sigma$ , there is some  $k > 0$  such that the random oracle  $\Sigma$ -way branching program time-space product of TTPRINT for CNF formulas with  $n$  clauses and  $\log(n) + \log \log(n)$  many variables is at least  $\Omega(n^2 / \log^k n)$ .*

**Proof.** We consider that the output of a machine computing NOE<sub>{0,1}</sub> can be a list  $L$  which contains the non-occurring elements of its input as well as any number of elements which are 0, as in Remark 3.3. We give a  $\text{poly}(\log(n))$ -time reduction with blowup  $\tilde{O}(n)$  from NON-OCCURRING ELEMENTS<sub>{0,1}</sub> to TTPRINT. That is, we will show that given a list  $L$  of  $n$  elements from  $[n]$ , we can produce a CNF formula whose truth table is a representation of a list  $L'$  of  $n$  strings each of  $\log(n)$  bits, where the  $i$ -th string in  $L'$  equals  $i$  if  $i \in [n] - L$ , otherwise the  $i$ -th string equals all-zeroes. Then, the lower bound for NON-OCCURRING ELEMENTS<sub>{0,1}</sub> will carry over to TTPRINT.

Given a list  $L \in \{1, \dots, n\}^n$ , we show how to efficiently construct a CNF formula  $F$  with  $\log(n) + \log \log(n)$  variables and  $O(n)$  clauses such that the truth table of  $F$  is the binary representation of the list of non-occurring elements of  $l$  separated by strings of 0's.

Denote the first  $\log(n)$  variables of  $F$  by  $x = x_1 \cdots x_{\log(n)}$ . Letting  $b \in \{0, 1\}^{\log(n)}$ , we make a clause  $C_b(x)$  expressing that  $x \neq b$ . In detail, suppose  $b$  is represented by the bit string  $b_1, \dots, b_{\log(n)}$ . Then

$$C_b(x) := ((x_1 \oplus b_1) \vee \dots \vee (x_{\log(n)} \oplus b_{\log(n)})).$$

Note for all  $i$ , we can think of  $C_b$  as containing the literal  $\bar{x}_i$  if  $b_i = 1$ , otherwise it contains the literal  $x_i$ .

Given a list  $L = \ell_1, \dots, \ell_n$  of elements of  $\{0, 1\}^{\log(n)}$ , we first construct a CNF formula  $F'(x)$  which says that  $x$  is a non-occurring element of  $\ell$ :

$$F'(x) := C_{\ell_1}(x) \wedge \dots \wedge C_{\ell_n}(x).$$

Suppose we can construct a CNF formula  $D(x, i)$  which is true exactly when the  $i$ -th bit of  $x$  is 1, and define our output formula to be

$$F(x, i) := F'(x) \wedge D(x, i)$$

where  $i = i_1 \cdots i_{\log \log(n)}$ . This  $F$  would have  $\log(n) + \log \log(n)$  variables, and its truth table in lexicographical order could be viewed as a list of  $n$  different  $\log(n)$ -bit strings, each of which are either a non-occurring element of  $\ell$ , or the all-zeroes string. (This would complete our reduction.)

We show how to construct such a  $D(x, i)$  with  $|x|$  clauses. For each variable of  $x$ , and  $j = 1, \dots, |x|$ , we construct a clause  $E_j$  which is true if and only if either  $i \neq j$  or  $x_j = 1$ . That is, we define  $E_j = ((i_1 \oplus j_1) \vee \dots \vee (i_{\log(|x|)} \oplus j_{\log(|x|)}) \vee x_j)$ . Then we can define

$$D(x, i) := \bigwedge_{j=1}^{|x|} E_j.$$

The final formula  $F(x_1, \dots, x_{\log(n)}, i_1, \dots, i_{\log \log(n)})$  is a width- $(\log(n)+1)$  CNF of  $O(n)$  clauses, and all of the clauses of  $F$  can be efficiently constructed in a local way. ◀

Using the lower bound on truth table printing, we can now show the lower bound for counting SAT assignments (Theorem 5). Again we do this by providing a  $\text{poly}(\log(n))$ -time reduction with blowup  $\tilde{O}(n)$  from one problem to another. We show how to modify a circuit  $C$  to efficiently produce another circuit  $C'$ , such that the bit representation of the number of SAT assignments of  $C$  equals the truth table of  $C$ .

► **Lemma 28.** *Let  $C$  be a circuit of size  $s$  with  $n$  input variables  $x_1, \dots, x_n$ . Then, there exists a circuit  $C'$  of size  $O(s + 2^n)$  with input variables  $x_1, \dots, x_n, y_1, \dots, y_{2^n}$  such that  $\#(C') = TT(C)$ . Moreover, there is an algorithm that given such a circuit and index  $i$  can produce the  $i$ -th bit of  $C'$  in time  $\tilde{O}(\log(s + n + i))$ .*

**Proof.** Let  $C$  be a circuit of size  $s$  with inputs  $x_1, \dots, x_n$ . First, we construct a new circuit  $D(x_1, \dots, x_n, y_1, \dots, y_{2^n})$  which outputs 1 if and only if  $y \leq 2^{\text{bin}(x)}$ , where  $\text{bin}(x)$  is the number in  $\{1, \dots, 2^n\}$  represented by the binary string  $x_1 \cdots x_n$ . Note that  $D$  can be constructed with  $O(2^n)$  gates (by standard arguments), and any particular gate of  $D$  can be constructed in  $\text{poly}(\log(n))$  time. Finally, we define the circuit  $C'$  on variables  $x_1, \dots, x_n, y_1, \dots, y_{2^n}$  by

$$C'(x, y) := C(x) \wedge D(x, y).$$

Observe that the number of assignments satisfying  $C'$  is  $\sum_{x \in \{0, 1\}^n} C(x) \cdot 2^{\text{bin}(x)}$ , as for each  $x \in \{0, 1\}^n$  there are  $2^{\text{bin}(x)}$  assignments to  $y \in \{0, 1\}^{2^n}$  such that  $C'(x, y) = 1$ . This is exactly the number represented in binary by  $TT(C)$ . ◀

From Lemma 28 and Theorem 4, we can conclude Lemma 29.

► **Lemma 29.** *For all finite  $\Sigma$ , there is some  $k > 0$  such that the random oracle  $\Sigma$ -way branching program time-space product of  $\#\text{CIRCUIT SAT}$  is at least  $\Omega(n^2/\log^k n)$ .*

From Lemma 29, we can conclude Lemma 30.

► **Lemma 30.** *For all finite  $\Sigma$ , there is some  $k > 0$  such that the random oracle  $\Sigma$ -way branching program time-space product of  $\#\text{3SAT}$  is at least  $\Omega(n^2/\log^k n)$ .*

**Proof.** The standard reduction from  $\#\text{CIRCUIT SAT}$  to  $\#\text{3SAT}$  can be implemented as an  $O(\log(n))$ -time reduction with blowup  $\tilde{O}(n)$ . ◀

From Lemmas 30 and 19, we can conclude Theorem 5.

► **Reminder of Theorem 5.** *For all finite  $\Sigma$ , there is some  $k > 0$  such that the random oracle  $\Sigma$ -way branching program time-space product of  $\#\text{2SAT}$  is at least  $\Omega(n^2/\log^k n)$ .*

**Proof.** Lemma 19 gives an  $O(\log^k(n))$  time reduction with blowup  $O(n\log^k(n))$  from  $\#\text{3SAT}$  to  $\#\text{2SAT}$ . Hence, by Lemmas 30 and 18, we can conclude that the random oracle  $\Sigma$ -way branching program time-space product of  $\#\text{2SAT}$  is at least  $\Omega(n^2/\log^k(n))$ . ◀

## 5 Conclusion

We extended a lower bound framework for branching programs, lifting it to random oracles. We demonstrated its utility for lower bound results by giving several unorthodox reductions which show sharp relationships between counting satisfying assignments, printing truth tables of CNFs, printing satisfying assignments, and evaluating circuits on given inputs. It would be interesting to find more applications of the encoding techniques used in our reductions. Perhaps they can be used to show lower bounds for other models, or better algorithms.

Another interesting direction would be to explore other lower bound methods, such as those for *decision* problems, and determine to what extent they can be “lifted” to lower bounds with random oracles. To give one tantalizing example, although it is known that deciding SAT requires  $n^{1.8}$  time on deterministic  $O(\log n)$ -space machines, and this paper shows that *printing* SAT assignments requires  $n^{2-o(1)}$  time on *randomized*  $O(\log n)$ -space machines with constant error probability, it is still open whether *deciding* SAT is in  $O(n)$  time and  $O(\log n)$  space on randomized machines with two-sided error(!). Perhaps it is easier to find bridges between function problems and decision problems when we consider efficient programs for NP problems (rather than decision problems in P).

Finally, our oracle model for branching programs is not the most general that one could imagine (although for the *random* oracle case, we believe it does not matter). One can define a sensible “extended oracle” model, where oracle queries can be as long as the height of the branching program. (This model is not very practical in a truly space-bounded setting, e.g., when we view a random oracle as a random hash function, but it would be interesting for lower bounds.)

At a high level, here is how such an “extended oracle” model can be defined. In each step, we allow the BP to output an “oracle character”  $\sigma$  from the input alphabet of the oracle (along with its usual outputs). Instead of labeling the query vertices of the BP with specific query strings (as in Definition 11), we instead label them with a special symbol  $Q$ . The  $Q$ -vertices still have two outgoing arcs for their yes/no query answers. However, each time a  $Q$ -vertex  $v$  is reached during a computation, the outgoing *yes* edge of  $v$  is now taken in the computation path if and only if the string of oracle characters  $y = \sigma_1 \cdots \sigma_t$  output since the

previous  $Q$ -vertex (or source node, if there is no previous  $Q$ -vertex) satisfies  $Q(y) = 1$ . One can think of this as allowing the BP “append-only” access to an arbitrarily long oracle tape, for which it can ask queries, and for which the oracle tape is reset to blank after each query (as in the oracle model of Ladner and Lynch [20]).

With the above model, we can ask queries whose length is only bounded by the height of the branching program (rather than the logarithm of its size). We strongly believe that our lower bounds also hold for random oracles in this more powerful model. The main conceptual bottleneck is that, unlike normal branching programs, we cannot easily partition these extended-oracle branching programs into short independent branching programs: the oracle queries have “memory” that can stretch all the way back to the source node. It seems likely that lower bounds based on Yao’s principle [29] can be extended to this model, but we have not yet confirmed this.

---

## References

- 1 Karl Abrahamson. Generalized string matching. *SIAM Journal on Computing*, 16(6):1039–1051, 1987.
- 2 Karl Abrahamson. Time-space tradeoffs for algebraic problems on general sequential machines. *Journal of Computer and System Sciences*, 43(2):269–289, 1991.
- 3 Miklós Ajtai. Determinism versus nondeterminism for linear time RAMs with memory restrictions. *Journal of Computer and System Sciences*, 65(1):2–37, 2002.
- 4 Miklós Ajtai. A Non-linear Time Lower Bound for Boolean Branching Programs. *Theory of Computing*, 1(8):149–176, 2005. doi:10.4086/toc.2005.v001a008.
- 5 Sanjeev Arora and Boaz Barak. *Computational Complexity: A Modern Approach*. Cambridge University Press, 2009.
- 6 David A. Mix Barrington and Pierre McKenzie. Oracle branching programs and Logspace versus P. *Information and Computation*, 95(1):96–115, 1991.
- 7 K. E. Batchner. Sorting Networks and Their Applications. In *Proceedings of the April 30–May 2, 1968, Spring Joint Computer Conference*, AFIPS ’68 (Spring), pages 307–314, 1968.
- 8 Paul Beame. A general sequential time-space tradeoff for finding unique elements. *SIAM Journal on Computing*, 20(2):270–277, 1991.
- 9 Paul Beame, Raphaël Clifford, and Widad Machmouchi. Element distinctness, frequency moments, and sliding windows. In *FOCS*, pages 290–299. IEEE, 2013.
- 10 Paul Beame, Thathachar S Jayram, and Michael Saks. Time-space tradeoffs for branching programs. *Journal of Computer and System Sciences*, 63(4):542–572, 2001.
- 11 Paul Beame, Michael Saks, Xiaodong Sun, and Erik Vee. Time-space trade-off lower bounds for randomized computation of decision problems. *JACM*, 50(2):154–195, 2003.
- 12 Paul Beame and Erik Vee. Time-space tradeoffs, multiparty communication complexity, and nearest-neighbor problems. In *STOC*, pages 688–697. ACM, 2002.
- 13 Allan Borodin and Stephen Cook. A time-space tradeoff for sorting on a general sequential model of computation. *SIAM Journal on Computing*, 11(2):287–297, 1982.
- 14 Samuel R. Buss and Ryan Williams. Limits on Alternation Trading Proofs for Time-Space Lower Bounds. *Computational Complexity*, 24(3):533–600, 2015.
- 15 Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. *Introduction to Algorithms*. MIT Press, 2009.
- 16 Devdatt P. Dubhashi and Alessandro Panconesi. *Concentration of measure for the analysis of randomized algorithms*. Cambridge University Press, 2009.
- 17 Lance Fortnow, Richard J. Lipton, Dieter van Melkebeek, and Anastasios Viglas. Time-space lower bounds for satisfiability. *JACM*, 52(6):835–865, 2005.



- 18 Etienne Grandjean. Linear Time Algorithms and NP-Complete Problems. *SIAM J. Comput.*, 23(3):573–597, 1994.
- 19 Christian Hoffmann. Exponential Time Complexity of Weighted Counting of Independent Sets. *CoRR*, abs/1007.1146, 2010. [arXiv:1007.1146](https://arxiv.org/abs/1007.1146).
- 20 Richard E Ladner and Nancy A Lynch. Relativization of questions about log space computability. *Mathematical Systems Theory*, 10(1):19–32, 1976.
- 21 Yishay Mansour, Noam Nisan, and Prasoos Tiwari. The computational complexity of universal hashing. *Theoretical Computer Science*, 107(1):121–133, 1993.
- 22 Jakob Pagter. On Ajtai’s Lower Bound Technique for R-way Branching Programs and the Hamming Distance Problem. *Chicago J. Theor. Comput. Sci.*, 2005.
- 23 Wolfgang J. Paul, Nicholas Pippenger, Endre Szemerédi, and William T. Trotter. On Determinism versus Non-Determinism and Related Problems (Preliminary Version). In *FOCS*, pages 429–438, 1983.
- 24 Rahul Santhanam. Lower bounds on the complexity of recognizing SAT by Turing machines. *Inf. Process. Lett.*, 79(5):243–247, 2001.
- 25 Martin Sauerhoff and Philipp Woelfel. Time-space tradeoff lower bounds for integer multiplication and graphs of arithmetic functions. In *STOC*, pages 186–195. ACM, 2003.
- 26 Dieter van Melkebeek. A survey of lower bounds for satisfiability and related problems. *Foundations and Trends in Theoretical Computer Science*, 2(3):197–303, 2007.
- 27 R. Ryan Williams. Time-Space Tradeoffs for Counting NP Solutions Modulo Integers. *Computational Complexity*, 17(2):179–219, 2008.
- 28 Ryan Williams. Nonuniform ACC Circuit Lower Bounds. *JACM*, 61(1):2, 2014.
- 29 Andrew C. Yao. Near-optimal time-space tradeoff for element distinctness. In *FOCS*, pages 91–97, 1988.
- 30 Yaacov Yesha. Time-space tradeoffs for matrix multiplication and the discrete Fourier transform on any general sequential random-access computer. *Journal of Computer and System Sciences*, 29(2):183–197, 1984.



# Random Projection in the Brain and Computation with Assemblies of Neurons

Christos H. Papadimitriou

Columbia University, USA  
christos@columbia.edu

Santosh S. Vempala

Georgia Tech, USA  
vempala@gatech.edu

---

## Abstract

---

It has been recently shown via simulations [8] that random projection followed by a *cap* operation (setting to one the  $k$  largest elements of a vector and everything else to zero), a map believed to be an important part of the insect olfactory system, has strong locality sensitivity properties. We calculate the asymptotic law whereby the overlap in the input vectors is conserved, verifying mathematically this empirical finding. We then focus on the far more complex homologous operation in the mammalian brain, the creation through successive projections and caps of an assembly (roughly, a set of excitatory neurons representing a memory or concept) in the presence of recurrent synapses and plasticity. After providing a careful definition of assemblies, we prove that the operation of assembly projection converges with high probability, over the randomness of synaptic connectivity, even if plasticity is relatively small (previous proofs relied on high plasticity). We also show that assembly projection has itself some locality preservation properties. Finally, we propose a large repertoire of assembly operations, including *associate*, *merge*, *reciprocal project*, and *append*, each of them both biologically plausible and consistent with what we know from experiments, and show that this computational system is capable of simulating, again with high probability, arbitrary computation in a quite natural way. We hope that this novel way of looking at brain computation, open-ended and based on reasonably mainstream ideas in neuroscience, may prove an attractive entry point for computer scientists to work on understanding the brain.

**2012 ACM Subject Classification** Theory of computation → Models of computation, Theory of computation → Randomness, geometry and discrete structures

**Keywords and phrases** Brain computation, random projection, assemblies, plasticity, memory, association

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2019.57

**Funding** This work was supported by NSF grants CCF-1563838, CCF-1819935, CCF-1763970, and CCF-1717349.

**Acknowledgements** Many thanks to Wolfgang Maass for many insightful discussions and exchanges during the early stages of our thinking in this area in general, and specifically about assembly operations, to Mike Collins for his insights regarding natural language in the human brain, and to Saket Navlakha for helpful comments on an early draft.

## 1 Introduction

The striking computational nature of the animal brain manifests itself even in the humblest circumstances. Flies sense odorants in their environment through specialized *olfactory*



© Christos H. Papadimitriou and Santosh S. Vempala;  
licensed under Creative Commons License CC-BY

10th Innovations in Theoretical Computer Science (ITCS 2019).

Editor: Avrim Blum; Article No. 57; pp. 57:1–57:19



Leibniz International Proceedings in Informatics

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

*receptor* neurons, of which there are roughly fifty different kinds. So, each smell is initially coded as a vector in 50 dimensions, where each coordinate is the level of activity of neurons of each kind. Then a remarkable thing happens: This vector undergoes a *random projection* – a familiar ingredient of many algorithms, especially in connection to learning [7, 2, 23, 1, 3] – to a higher dimensional space. There is a  $50 \times 2000$  sparse, and by all evidence [6] random, bipartite graph of synapses projecting the 50 kinds of olfactory receptors to a population of 2000 neurons called *Kenyon cells*. Next, the resulting 2000-dimensional vector of synaptic inputs undergoes an operation that is routine in neural systems: The activity of the Kenyon cells excites an inhibitory neuron, and the resulting activity of this neuron, at equilibrium, has the effect of increasing everybody’s membrane potential, “turning off” all but roughly the 100 most active cells. We call this operation *cap*; it is also known as *k winners take all*, in this case with  $k = 100$ .

In a recent paper [8] it was shown empirically that this mapping, random projection followed by *cap*, has strong *locality sensitivity* properties (and therefore preserves similarity of smells, presumably to the animal’s advantage), in fact outperforming in simulations certain variants of locality-sensitive hashing<sup>1</sup>. One of our results in this paper puts some mathematical teeth to this interesting empirical observation: We prove that if two binary vectors of the same sparsity overlap in a fraction  $\alpha$  of their entries, and both undergo random projection to  $n$  dimensions followed by  $k$ -*cap*, then the two results will overlap in a fraction of about  $(\frac{k}{n})^{\frac{1-\alpha}{1+\alpha}}$  (Theorem 1). For the small numbers of the insect brain ( $\frac{n}{k} \approx \frac{2000}{100}$ ), this is substantial overlap that helps explain the empirical findings in [8] (see Figure 1).

In the mammalian brain numbers get roughly three orders of magnitude higher, and yet something similar seems to happen. Importantly, there is strong *recurrent synaptic connectivity* between excitatory neurons; that is, the random graph is now not just a directed bipartite graph, but the union of a bipartite directed graph and a non-bipartite directed graph interconnecting the receiving side (in contrast, synapses between the fly’s Kenyon cells, if any, play no role there). In mammals, the random projection and *cap* operation does take place, but it is only the first step of a complex and sophisticated process, culminating in the creation of an *assembly of neurons*.

**Assemblies.** Already in 1949, neuroscience pioneer Donald Hebb predicted that memories and concepts are represented by tightly connected sets of neurons he called *assemblies*, whose near-simultaneous firing is tantamount to these concepts being thought about. During the last decade, it has been established experimentally [13, 14, 19], see also the survey [5], that such near-simultaneous firing of stable sets of neurons is an important part of the way the brain works. Assemblies have been hypothesized to underlie many of the higher cognitive operations in mammals, such as memory, reasoning, language, planning, etc., and yet, the way and manner in which this happens has not begun to be articulated; the computational framework of this paper is a first attempt at understanding how assemblies of neurons can carry out computation.

**In our framework.** In our framework, the brain is divided into a bounded number of *brain areas*. Each brain area contains a number of excitatory neurons denoted by  $n$ ; there are of course other neurons as well, for instance see the discussion on inhibition below. These excitatory neurons are interconnected in a sparse directed  $G_{n,p}$  graph. Pairs of brain areas

---

<sup>1</sup> As Alex Andoni notes (private communication, 2018), this is not true of the more advanced versions of LSH.

may also be connected, in one or both directions, through bipartite directed  $G_{n,p}$  graphs<sup>2</sup>.

Finally, the other two important aspects of our model are *cap* and *plasticity*. We assume that neurons fire – or do not – in discrete time steps (a very convenient and unrealistic assumption, which however does not interfere much with the rest of our framework). At each time and each brain area, the  $k$  out of  $n$  neurons that have largest synaptic input fire. That is, at time  $t$  for each neuron we add together the weights of the incoming synapses that originate in neurons (in the same or different area) which fired the previous time  $t - 1$ , and select the  $k$  neurons out of the  $n$  in the brain area that have the largest sums. These are the neurons in the area that will fire at time  $t$ . The  $k$ -cap process is a simplification and approximation of the reality of *inhibition*, whereby an independent population of inhibitory neurons cause the excitatory neurons to have high enough membrane potential that an equilibrium at  $k$  firing neurons is quickly reached. Finally, *plasticity*: we assume that if there is a synapse from neuron  $i$  to neuron  $j$ , and neuron  $i$  fires at time  $t$  while neuron  $j$  at  $t + 1$ , the weight of the synapse is increased by a factor of  $1 + \beta$  with  $\beta > 0$ ; synaptic weights start at one, say<sup>3</sup>. Thus, the key parameters of our model are  $n, k, p, \beta$ , whose indicative intended values for the mammalian brain are, respectively,  $10^7, 10^4, 10^{-3} - 10^{-2}, 10^{-1}$ .

**Defining Assemblies.** An assembly is of course a set of neurons, in our framework all belonging to the same brain area. In past theoretical work [17] this is exactly how they were defined, a set of  $k$  neurons firing simultaneously. It is a highly interconnected set to ensure *stability*, that is, if enough neurons in it fire then soon all of them will<sup>4</sup> – and one of the main points of [17] was that there is a biologically plausible algorithm for selecting such a highly connected set of neurons in a sparse  $G_{n,p}$  graph. These neurons might be poised to fire in a particular pattern, not necessarily all simultaneously as was assumed in [17] – and indeed, in our simulations, as well as in the literature on assembly simulations, one does see nontrivial patterns of firing. We believe the right way to define assemblies is as *distributions over the set of neurons in a Brain area whose support has size at most a fixed multiple of the cap size  $k$* .

**Projection.** The most basic operation of assemblies is what we call *projection* – this is how assemblies are created and, once created, *copied* to other brain areas for further use. Assembly projection has been conjectured for a long time and has been established in several simulation papers [20, 18] and recently analytically proved [17] for a range of parameters. An assembly  $x$  in area  $A$  can project to a different area  $B$ , to which  $A$  has ample connectivity, creating a new assembly  $y$ ; this operation is denoted  $\text{project}(x, B, y)$ . If in the future  $x$  is activated,  $y$  will follow suit; we say that  $x = \text{parent}(y)$ . We show that the operation  $\text{project}(x, B, y)$  is carried out by assembly  $A$  simply *firing for a small number of steps*<sup>5</sup>. Once an assembly  $x$  has been created, its area is implicit, denoted by  $\text{area}(x)$ . To create

<sup>2</sup> See [17] for a technical discussion of *synaptic biases*, departures from the  $G_{n,p}$  model noted in experiments, and the reasons why they may provide further support for the assembly hypothesis. We do not pursue this direction in the present paper.

<sup>3</sup> There should also be a process of *homeostasis* which, at a slower time scale, keeps the sum of all weights from growing; but this aspect of the model, taken up in Section 5, does not affect the relative ordering of synaptic weights or sums thereof.

<sup>4</sup> This is one of the many important differences between this work and Valiant’s pioneering theory of *items* from the 1990s [21, 22]

<sup>5</sup>  $\text{project}(x, B, y)$  may seem superficially equivalent to an assignment  $x = y$  in a programming language – except that, after such an assignment, variables  $x$  and  $y$  go on to live largely independent lives, whereas in assemblies  $x$  retains power over  $y$ , while  $y$  can only exist through  $x$ .

an altogether new assembly  $y$  by  $\text{project}(x, B, y)$ ,  $x$  must be a “proto-assembly,” a set of neurons coding a world experience and residing at some higher area of the sensory cortex (such as the area IT of the visual cortex where whole objects are represented), projected to a non-sensory area admitting new assemblies (typically the hippocampus). One of our main results in this paper (Theorem 3) is that projection indeed works as described – with high probability, of course, with randomness supplied by the graph, and in fact for quite low plasticity.

The projection process is quite intricate. It starts with the random projection plus  $k$ -cap described early in this introduction, creating a set of neurons that we call  $A_1$ , namely, the cells that happen to have the largest synaptic input from the projecting assembly  $x$ . We *assume* that the synaptic input of a neuron from assembly  $x$  is a Bernoulli random variable with parameters  $k, p$  and  $n$  samples. Notice also that, after the first round, the synapses between  $x$  and  $A_1$  have been boosted by plasticity. As the projecting assembly keeps firing, cap will select the set of neurons  $A_2$  that have highest *combined* synaptic input from  $x$  and  $A_1$ , and these will include two kinds of cells: the *core* neurons in  $A_1 \cap A_2$ , and new winners from outside  $A_1$ . What fraction of  $A_1$  will become core? This is an important parameter of the situation, and we call it  $\lambda$ . To compute it, we set up an algebraic equation of Bernoulli expectations; as the expectation of a Bernoulli quantile depends explicitly on the fraction of winners, and concentration is strong, we can set up the equation and solve it in the “high probability” sense. For the parameter range of interest,  $\lambda$  is about half. Notice that, after this step, all synapses from  $x$  and  $A_1$  to  $A_2$  are boosted by plasticity.

Then the process is repeated,  $A_3, A_4, \dots, A_t, \dots$ , and we wish to show that  $|B^*| = |\bigcup_t A_t|$  converges to some finite multiple of  $k$  (recall that this is our definition of an assembly). That is, eventually there will be a time after which there are no first-time winners. Unfortunately our already complicated Bernoulli analysis is no longer an option, for a variety of reasons. First, at time  $t$  the number of types of neurons grows exponentially with  $t$ : the type of each neuron is the set of  $\tau$ 's for which the neuron was in  $A_\tau$ . In addition, the distribution of the synaptic input of neurons with complex type is not Bernoulli, because of conditioning. Instead, we resort to classifying each neuron by its *rough type at time  $t$* , which is the number of *consecutive* times  $\tau$  leading to  $t - 1$  during which the neuron was in  $A_\tau$ . A crucial lemma states that the probability that the run will end at time  $t$  and the neuron will find itself outside  $A_t$  decreases exponentially with the length of the run (that is to say, the neuron's rough type), and in fact uniformly in  $t$ . Convergence to a union size that is a multiple of  $k$  (with a multiplier that is, naturally, a steeply increasing function of  $\frac{1}{\beta}$ ) follows (Theorem 3).

The proof is quite a bit easier in the *high plasticity regime* defined by  $\beta > \sqrt{\frac{(1-p)\ln n}{pk}}$ , in which case convergence is stronger in that the sequence  $A_t$  itself converges in finitely many steps (as indicated in [17]).

**Operations on Assemblies.** What is the right scale for understanding computation in the brain? We suspect that assemblies may underlie an important and powerful mode of brain computation, complementary to the computation involved in the processing of sensory input – heretofore the main focus of neuroscience. Such computation would encompass memory recall and association, deduction and reasoning, generating and parsing natural language, generating and manipulating stories and plans, even math. It happens at a level of abstraction intermediate between individual neurons and synapses at the lowest level, and whole brain computation at the highest; it is far more expressive than the latter, and much less cumbersome to describe than the former. In our quest to understand the full power of this mode of computation, in Section 5 we identify a repertoire of additional operations on

assemblies, beyond projection. We only seek operations that are “realistic” in the following two orthogonal senses: (a) operations for which there is experimental evidence, in the sense that their existence would help explain extant experimental data, and which could possibly be themselves tested experimentally; and (b) operations which are in addition *plausible*, shown (analytically if at all possible, otherwise through simulations) to be realizable at the level of neurons and synapses in our framework. That is to say, each assembly operation must be “compiled down” to the level of neurons and synapses. Our list of operations includes, besides projection: *association*, in which two assemblies in the same area increase their intersection to reflect conceptual or statistical affinity – there is extensive experimental evidence for this operation, see [17] for an extensive discussion; *merge*, in which two assemblies from two different areas project to *the same new assembly* in a third area, an operation that seems important for processing syntax in natural language; *reciprocal project* (like project, except that the projected assembly is able to activate the original one, in addition to vice-versa); and *append*, an operation useful for creating and maintaining sequences. There are also several *control operations* allowing one to *read* the information of assembly activity in specific areas, or *disable* synaptic connectivity between areas – ultimately, to *write simple programs*. We show that this repertoire of assembly operations constitutes a programming system<sup>6</sup> which can simulate arbitrary computation in a way that is quite natural (Theorem 4). The point of this exercise is to demonstrate the power of this basis of primitives, not to hypothesize that the brain must function exactly this way.

## Related work

Our work on assemblies is superficially related to (and was undoubtedly inspired by) Valiant’s theory of *items*. There are stark contrasts between the two approaches: Assemblies are hypothesized to be densely connected, a requirement that makes their creation challenging, while items are ransom sets of neurons. And we believe that our model is far closer to the realities of the brain, as they are known now, than Valiant’s; for one key difference, Valiant assumes plasticity (change in synaptic weights) to be arbitrarily programmable at the post-synaptic site, while we assume a very simple implementation of Hebb’s rule. With this model we are able to address the problem of how the brain creates similar representations for similar stimuli.

Our earlier work on assemblies established experimentally the plausibility of projection and association [20], and theoretically so by relying on very high plasticity [17]. In this paper, we attack analytically the more realistic and considerably more challenging regime of small plasticity.

## 2 Model

We assume a finite number of brain areas, denoted by  $A, B, \dots$ . Each brain area is a weighted directed graph whose vertices are  $n$  (think of  $n$  as  $10^6$  or  $10^7$ ) excitatory neurons, and whose edges are synapses between neurons; the positive weights vary dynamically through plasticity, see below. We assume that the edges are drawn from a  $G_{n,p}$  distribution. That is, we assume that the probability of any edge is  $p$  and edges are chosen independently. In addition, between certain ordered pairs of areas  $(A, B)$  there is a  $G_{n,p}$  directed bipartite graph from nodes of  $A$  to nodes of  $B$ . In other words, there is a finite directed graph with the areas as

<sup>6</sup> Which, to our credit, we refrained from dubbing “Assembly Language”...

nodes, determining whether the two areas have synaptic connections. We assume that there is a mechanism to *disable* the synaptic connections between two areas  $A$  and  $B$  at any time.

We assume that events happen in discrete time steps (think of each step as about 20 ms). At each step  $t$ , every neuron  $i$  in every area  $A$  may or may not *fire*. Whether  $i$  fires depends on its *synaptic input* at time  $t$ . This is defined the sum over all neurons  $j$  that have synapses  $(j, i)$  (note that  $j$  can be either in area  $A$  or in an area  $B$  that does have synapses into  $A$  that are not disabled at time  $t$ ). Denote this quantity as  $SI(j)$ . We assume that neuron  $i$  in area  $A$  fires at time  $t$  if and only if  $|\{j \in A : SI(j) \geq SI(i)\}| < k$ , where  $k$  is a key parameter of the model (think of it as roughly  $\sqrt{n}$ ). We call the set of neurons firing at a time  $t$  the *cap* of the area. The cap is a mathematically tractable way of capturing the important process of *inhibition*, whereby inhibitory neurons in an area (typically outnumbering excitatory ones) are excited by the firing of excitatory neurons in the area, and in response fire, preventing some excitatory neurons from further firing, and eventually reaching an equilibrium (called the *E-I balance* in the literature). Here we model this equilibrium by a constant  $k$  and ignore the transient.

The other important ingredient of our model is plasticity: We assume that if there is a synapse with weight  $w$  from neuron  $i$  to neuron  $j$  (either in the same area, or in another area with enabled synapses), and it so happens that  $i$  fires in time  $t - 1$  and  $j$  fires in time  $t$ , then the weight of synapse  $ij$  is in time  $t + 1$  equal to  $w(1 + \beta)$ , where  $\beta$  (think of it as between 0 and 1, realistically at the lower end of this) is the plasticity coefficient. Plasticity is a very complex phenomenon with many important aspects and cases, but we feel that this simple rule (corresponding to Hebb's "fire together wire together" maxim) captures the essence of the matter reasonably well.

We shall elaborate certain further aspects of our model in the section on assembly operations.

### 3 The Overlap of Projections

In this and the next section we analyze how assemblies can be formed in our model. We assume that there is a *stimulus*  $A$  of  $k$  neurons firing in an area, with enabled synaptic projections to another area, where the assembly will be formed. We start with the simple case (modeling the insect brain) where  $A$  fires only once, forming the cap in the downstream area denoted  $\text{cap}(A)$ , and analyze how the overlap of two stimuli  $A$  and  $B$  is maintained in the process; note that here recurrent connections and plasticity do not get involved, and the weights can be thought to be one. The following observation will be useful: conditioning on a neuron not making it to a cap cannot increase its cap probability for future steps.

► **Lemma 1.** *Let  $A, B$  be two stimuli. Then for any node  $i \in V$ ,*

$$\Pr(i \in \text{cap}(B) \mid i \notin \text{cap}(A)) \leq \Pr(i \in \text{cap}(B)) = \frac{k}{n}$$

where the probability is over the randomness of the graph.

Also, we will need the following well-known bound on the Gaussian tail.

► **Lemma 2 (Gaussian tail).** *For  $x \sim N(0, 1)$  and  $t > 0$ ,*

$$\frac{1}{\sqrt{2\pi}} \left( \frac{1}{t} - \frac{1}{t^3} \right) \exp(-t^2/2) \leq \Pr(x \geq t) \leq \frac{1}{\sqrt{2\pi}t} \exp(-t^2/2).$$

Now we state and prove our quantitative assessment of the locality sensitivity properties of the insect olfactory map pointed out empirically in [8].

► **Theorem 3.** *The expected overlap of the caps two stimuli that overlap in an  $\alpha$  fraction of their nodes is*

$$\frac{|\text{cap}(A) \cap \text{cap}(B)|}{k} \gtrsim \frac{1}{(\ln(n/k))^{\frac{\alpha}{1+\alpha}}} \left(\frac{k}{n}\right)^{\frac{1-\alpha}{1+\alpha}}.$$

**Proof.** We bound the probability that any neuron  $i$  is in the cap of both  $A$  and  $B$ . For this, let  $x_i, y_i, z_i$  be the total input to node  $i \in V$  from  $A \setminus B, A \cap B$  and  $B \setminus A$ . Then  $x_i, z_i \sim N((1-\alpha)kp, (1-\alpha)kp(1-p))$  and  $y_i \sim N(\alpha kp, \alpha kp(1-p))$ . Then, using the independence of  $x_i + y_i$  and  $z_i + y_i$  given  $y_i$ ,

$$\begin{aligned} & \Pr i \in \text{cap}(A) \cap \text{cap}(B) \\ &= \int \int \int \chi(x_i + y_i \in \text{top } k \text{ of } \{x_j + y_j\} \text{ and } z_i + y_i \in \text{top } k \text{ of } \{z_j + y_j\}) d\gamma(x) d\gamma(z) d\gamma(y) \\ &= \int \int \int \chi(x_i + y_i \in \text{top } k \text{ of } \{x_j + y_j\} | y) \chi(z_i + y_i \in \text{top } k \text{ of } \{z_j + y_j\} | y) d\gamma(x) d\gamma(z) d\gamma(y) \\ &\geq \int \left( \int \chi(x_i + y_i \in \text{top } k \text{ of } \{x_j + y_j\} | y) d\gamma(x) \right)^2 d\gamma(y) \\ &\geq \int_{y_i} [\Pr(x_i \geq -y_i + kp + t | y_i)]^2 d\gamma(y_i). \end{aligned}$$

The last step above is the simple observation that a random draw  $x_i + y_i$  from  $N(kp, kp(1-p))$  is, with constant probability, in the top  $k$  of  $n$  iid draws from the same distribution if  $x_i + y_i \geq \mathbb{E}(x_i + y_i) + t$  where  $\Pr(x_i + y_i \geq t) \geq k/n$ . The tail bound below shows that

$$t \sim \sqrt{(2 \ln(n/k) - \ln(2 \ln(n/k)))kp}.$$

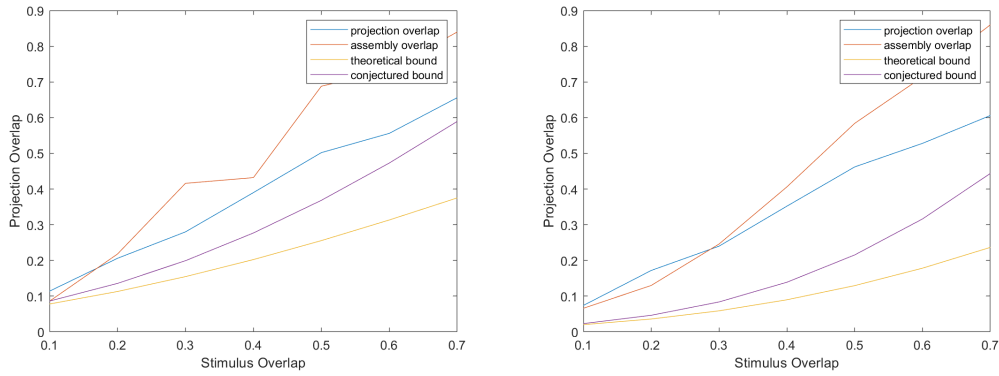
For convenience, we shift the distributions of  $x_i, y_i$  to  $\bar{x} = (x - (1-\alpha)kp)/kp$  and  $\bar{y} = (y - \alpha kp)/kp$  so that  $\bar{x} \sim N(0, (1-\alpha))$  and  $\bar{y} \sim N(0, \alpha)$ . For  $x \sim N(0, 1)$ , we will use the tail bound in Lemma 2:

$$\frac{1}{\sqrt{2\pi}} \left( \frac{1}{t} - \frac{1}{t^3} \right) \exp(-t^2/2) \leq \Pr(x \geq t) \leq \frac{1}{\sqrt{2\pi}t} \exp(-t^2/2).$$

Thus, for any  $\alpha < 1$ ,

$$\begin{aligned} & \Pr(i \in \text{cap}(A) \cap \text{cap}(B)) \\ &\geq \int_{\bar{y}} \Pr(\bar{x} \geq -\bar{y} + t)^2 d\gamma(\bar{y}) \\ &\geq \int_{\bar{y}} \frac{1}{2\pi(1-\alpha)} \min \left\{ \frac{1-\alpha}{(t-\bar{y})^2}, 1-\alpha \right\} \exp \left( -2 \frac{(t-\bar{y})^2}{2(1-\alpha)} \right) \frac{1}{\sqrt{2\pi\alpha}} \exp \left( -\frac{\bar{y}^2}{2\alpha} \right) d\bar{y} \\ &\geq \left( \frac{1}{2\pi t^{2/(1+\alpha)}} \exp \left( -\frac{t^2}{1+\alpha} \right) \right) \int_{\bar{y}} \frac{t^{2/(1+\alpha)}}{\sqrt{2\pi\alpha}} \min \left\{ \frac{1}{(t-\bar{y})^2}, 1 \right\} \exp \left( -\frac{(\bar{y} - \frac{2\alpha}{(1+\alpha)}t)^2}{2\alpha(1-\alpha)/(1+\alpha)} \right) d\bar{y} \\ &\geq \sqrt{\frac{1-\alpha}{1+\alpha}} \left(\frac{k}{n}\right)^{\frac{2}{1+\alpha}} \frac{1}{t^{2\alpha/(1+\alpha)}} \int_y \frac{\min \left\{ \frac{1}{\left(\frac{1-\alpha}{1+\alpha} - \frac{y}{t}\right)^2}, 1 \right\}}{\sqrt{2\pi\alpha(1-\alpha)/(1+\alpha)}} \exp \left( -\frac{y^2}{2\alpha(1-\alpha)/(1+\alpha)} \right) dy \\ &\geq \sqrt{\frac{1-\alpha}{1+\alpha}} \left(\frac{k}{n}\right)^{\frac{2}{1+\alpha}} \frac{1}{t^{2\alpha/(1+\alpha)}} \int_y \frac{1}{\sqrt{2\pi}} \min \left\{ \frac{1}{\left(\frac{1-\alpha}{1+\alpha} - \frac{y}{t} \sqrt{\frac{\alpha(1-\alpha)}{1+\alpha}}\right)^2}, 1 \right\} \exp \left( -\frac{y^2}{2} \right) dy \\ &\geq \frac{\sqrt{\frac{1-\alpha}{1+\alpha}}}{(2 \ln(n/k))^{\alpha/(1+\alpha)}} \left(\frac{k}{n}\right)^{\frac{2}{1+\alpha}}. \end{aligned}$$





■ **Figure 1** The first figure is with  $n = 2000, k = 100$  and the second with  $n = 10000, k = 100$ ; each empirical plot is the average of 5 independent trials. For the assembly creation we used plasticity of  $\beta = 0.1$ . The theoretical bound plotted is  $(k/n)^{(1-\alpha)/(1+\alpha)} / \ln(n/k)^{\alpha/(1+\alpha)}$ , while the conjectured bound is the same without the log factor.

Thus the expected fraction of overlap is this probability times  $n$  divided by  $k$ , i.e.,

$$\Omega \left( \frac{1}{(\ln(n/k))^{\frac{\alpha}{1+\alpha}}} \left( \frac{k}{n} \right)^{\frac{2}{1+\alpha}} \frac{n}{k} \right) = \Omega \left( \frac{1}{(\ln(n/k))^{\frac{\alpha}{1+\alpha}}} \left( \frac{k}{n} \right)^{\frac{1-\alpha}{1+\alpha}} \right). \quad \blacktriangleleft$$

It seems that the steps in this proof, including the suppression of constants in the end, are quite parsimonious, in that the stated lower bound is not very far from the truth. In Figure 1 we compare our bound with simulations of the map for various values of  $\alpha$  and with  $n/k = 2000/100 = 20$  (the values that pertain to insect olfaction) and  $n = 10^4, k = 100$ , and also to our bound without the logarithmic factor.

#### 4 Bounding the Support of an Assembly

In this section we turn to assemblies in the mammalian brain, in which recurrent synapses and plasticity become important. We assume that a stimulus consisting of  $k \geq \sqrt{n}$  neurons in an upstream area fires repeatedly. The cap at  $t = 1$ , denoted  $A_1$ , which was analyzed in the previous section, is only the preamble of a complex process. At  $t = 2$  the stimulus fires again, and now the area receives combined input from the stimulus *and* from  $A_1$ . A cap denoted  $A_2$  will be formed, probably containing a considerable part of  $A_1$  but also *first-timers* (by which we mean, neurons not heretofore participating in any cap). Meanwhile, plasticity has changed the weights. The process is repeated a number of times, with new winners displacing some past winners from the new cap, while plasticity acts in a stabilizing way. Convergence – that is,  $A_t = A$  for all  $t > t_0$  – cannot be guaranteed with high probability (experiments show some periodic-like movement of neurons, without any new first-timers). The interesting question is, will the process converge, in that after some point and after there will be no new winners? (Recall that this is what we mean by an assembly, a set of neurons of size a small multiple of  $k$  firing in a pattern.). If so, we are interested in the size of the assembly's *support*, the union of all the  $A_t$ s. The bound on the support depends crucially on the plasticity parameter  $\beta$ , with high plasticity leading to small support (close to the cap size  $k$ ) but even very small positive plasticity leading to bounded support size (a fact that is harder to prove). We denote by  $A^*$  the union of  $A_0, A_1, A_2, \dots$

► **Theorem 4** (High Plasticity). *Assume that the plasticity parameter  $\beta \geq \beta_0 = \frac{(\sqrt{2}-1)\sqrt{\ln n} + \sqrt{2}}{\sqrt{pk} + \sqrt{\ln n}}$ . Then WHP the total support of the assembly can be bounded as*

$$|A^*| \leq k \frac{1}{1 - \exp(-(\frac{\beta}{\beta_0})^2)} \leq k + O\left(\frac{\ln n}{p\beta^2}\right).$$

**Proof.** Let  $\mu_1 = 1, \mu_2, \dots, \mu_t, \dots$  be the fraction of first-timers in the cap at step  $t$ . The process stabilizes when  $\mu_t < 1/k$ . Using the tail bound of the Gaussian, since the new winners must be in the top  $\mu_t k$  of remaining  $n - k \sim n$  neurons, the activation threshold at step  $t$  is therefore very close to

$$C_1 = pk + \sqrt{2pk \ln \frac{n}{k}}, \quad C_t = 2pk + 2\sqrt{pk \ln \frac{n}{\mu_t k}} \text{ for } t \geq 2.$$

Note that the mean term is  $pk$  for the first step and  $2pk$  for all subsequent steps since the number of neurons firing is the  $k$  stimulus ones plus  $k$  from the brain area.

First consider a neuron that make it to the first cap. To bound the probability that that it will remain in the next cap, we note that at this point, the total activation from the input synapses is at least  $(1 + \beta)C_1$  and from the recurrent synapses it is at least  $X$  where  $X \sim N(pk, p(1-p)k)$  is the signal from the recurrent synapses coming from nodes in the first cap. In order for a node to remain in the next cap, we need that

$$(1 + \beta)C_1 + pk + X \geq C_2$$

where now  $X \sim N(0, p(1-p)k)$ . Substituting for  $C_1, C_2$ , and using  $L = 2 \ln(n/k)$ , and  $\mu$  as the fraction of first-timers in the second cap, we have

$$\begin{aligned} \Pr(j \in C_2 | j \in C_1) = 1 - \mu &\geq \Pr(X \geq -\beta pk - (1 + \beta)\sqrt{pkL} + \sqrt{2pk(L + 2 \ln(1/\mu))}) \\ &\geq \Pr(X \geq -\beta\sqrt{pk} + \sqrt{2(L + \ln(1/\mu))} - (1 + \beta)\sqrt{L}) \\ &\quad \text{rescaling so that } X \sim N(0, 1). \\ &\gtrsim 1 - \exp\left\{-\left(\beta\sqrt{pk} + (1 + \beta)\sqrt{L} - \sqrt{2(L + \ln(1/\mu))}\right)^2/2\right\}. \end{aligned}$$

In other words,

$$\sqrt{2 \ln(1/\mu)} \leq \beta\sqrt{pk} + (1 + \beta)\sqrt{L} - \sqrt{2(L + \ln(1/\mu))}.$$

Now setting

$$\beta \geq \beta_0 = \frac{(\sqrt{2}-1)\sqrt{L} + \sqrt{2}}{\sqrt{pk} + \sqrt{L}}$$

gives  $\mu < 1/e$ , i.e., the overlap with the next cap is at least a  $1 - (1/e)$  fraction. The probability of remaining in the cap rapidly increases with the number of consecutive times a neuron stays in the cap. To see this, suppose neuron  $j$  enters the cap for the first time at time  $t$ , by exceeding the threshold  $C_t$  and stays for  $i$  consecutive caps (including  $C_t$ ). The, to stay in the next cap, it suffices that

$$(1 + \beta)^i C_1 + pk + X \geq C_{i+1}$$

where  $X \sim (0, p(1-p)k)$ . Then, rescaling so  $X \sim N(0, 1)$ ,

$$\begin{aligned} \Pr(j \in C_{i+1} | j \in C_1) &= 1 - \mu \\ &\geq \Pr(X \geq (1 - (1 + \beta)^i)\sqrt{pk} - (1 + \beta)^i\sqrt{L} + \sqrt{2(L + 2 \ln(1/\mu))}) \\ &\gtrsim 1 - \exp\left\{-\left(i\beta\sqrt{pk} + (1 + i\beta)\sqrt{L} - \sqrt{2(L + \ln(1/\mu))}\right)^2/2\right\}. \end{aligned}$$

## 57:10 Computation with Assemblies

Rewriting,

$$\sqrt{2\ln(1/\mu)} + \sqrt{2(L + \ln(1/\mu))} - \sqrt{L} \leq i\beta(\sqrt{pk} + \sqrt{L})$$

or

$$\beta \geq \frac{1}{i} \cdot \frac{\sqrt{2\ln(1/\mu)} + \sqrt{2(L + \ln(1/\mu))} - \sqrt{L}}{(\sqrt{pk} + \sqrt{L})}$$

which is less than  $\beta_0$  for  $\mu = e^{-i^2}$ .

Next we consider a new first time winner in round  $t$ . In order for this neuron to make it to the cap at time  $t + 1$ , we need that

$$(1 + \beta) \frac{(2 - \mu)}{2} C_t + \mu pk + X \geq C_{t+1}$$

where  $\mu = \mu_{t+1}$  is the fraction of newcomers in the next cap and  $X \sim N(0, \mu p(1 - p)k)$ . Rescaling so that  $X \sim N(0, \mu)$ , we have  $\Pr(j \in C_{t+1} | j \in C_t)$  is

$$1 - \mu \geq \Pr(X \geq -\beta(1 - \frac{\mu}{2})2\sqrt{pk} - (1 + \beta)(1 - \frac{\mu}{2})\sqrt{2(L + \ln(1/\mu_t))} + \sqrt{2(L + \ln(1/\mu))})$$

Using the tail bound and rewriting as before, we have

$$\beta \geq \frac{2\ln(1/\mu) + \frac{\mu}{2}\sqrt{2(L + \ln(1/\mu_t))} + \frac{\ln(\mu_t/\mu)}{L}}{(1 - \frac{\mu}{2})(2\sqrt{pk} + \sqrt{2(L + \ln(1/\mu_t))})}$$

which is less than  $\beta_0$  for  $\mu = \mu_t/e$ . In other words, the  $\beta$  threshold to do this and ensure that  $\mu$  drops by a constant factor is lower than the threshold  $\beta_0$  for the first step. Finally, as before, the probability of staying in the cap increases rapidly with the length of the neurons' winning streak.

If  $\beta \geq \beta_0$ , then  $\mu_t$  drops off exponentially. i.e., the probability of leaving the cap once in the cap for  $i$  consecutive times  $1 - p_i^t$  drops off exponentially. Using these facts, we get

► **Claim 1.**

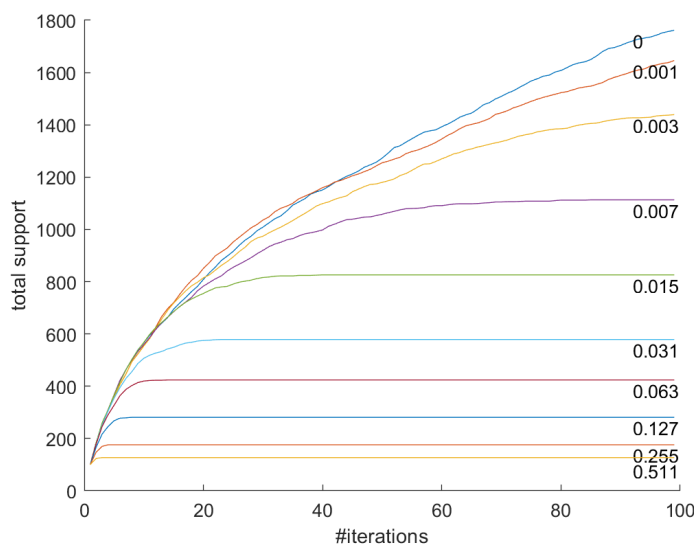
$$\prod_{i \geq 1} p_i \geq \prod_{i \geq 1} (1 - \exp(-i^2(\frac{\beta}{\beta_0})^2)) \geq \frac{1}{2}.$$

The claim gives a lower bound on the probability that a neuron that makes it to a cap for the first time remains in the cap for all future times. As a result, each neuron that makes it a cap for the first time has a probability of at least  $q = 1 - \exp(-(\frac{\beta}{\beta_0})^2)$  of remaining in all future caps. Thus, the total support of all caps together is at most  $k/q$  in expectation. This completes the proof of the theorem. ◀

We now turn to the regime of low plasticity, including zero plasticity. The bounds here will be higher asymptotically, as reflected also in our experiments (see Figure 2). We note however that for parameter ranges of interest for the brain, e.g.,  $n = 10^6, k = 10^3$ ,

$$\left(\frac{n}{k}\right)^{1/4} < \ln(n/k).$$

The guarantees below are meaningful and nontrivial only when  $k$  is sufficiently large as a function of  $n$ .



■ **Figure 2** The total support size at different values of plasticity  $\beta$  ranging from 0 to just over 0.5 for a random network with  $n = 10^4$  neurons, edge probability  $p = 0.01$  and assembly size  $k = 100$ . The  $x$  axis is the number of iterations.

► **Theorem 5 (Low Plasticity).** *Let a network with  $n$  nodes have edge density  $p$ , plasticity parameter  $\beta$ , and cap size  $k \geq \sqrt{n}$ . For a sequence of caps  $A_0, A_1, A_2, \dots, A_t, \dots$ , let  $A^*$  be their union. Denote  $\mu = \sqrt{k/n}$ . Then,*

1. for  $\beta = 0$ ,

$$\mathbb{E}(|A^*|) \leq k \left(\frac{1}{\mu}\right)^{\frac{1}{\mu}}.$$

2. for  $\beta > 0$ ,

$$\mathbb{E}(|A^*|) \leq k \left(\frac{1}{\mu}\right)^{\frac{1}{2\beta}}.$$

**Proof.** For the first part, let  $\mu_0, \mu_1, \dots, \mu_t, \dots$  be defined as  $\mu_0 = 0$  and

$$\mu_t = \frac{|A_t \cap A_{t-1}|}{k},$$

the fraction of the cap that persists to the next step.

We will show that the expected values of  $\mu_t$  form an increasing sequence and give a recursive lower bound. To get a lower bound on  $\mu_1$ , for a neuron  $j$ , let  $x$  be the total signal from the stimulus and  $y$  from  $A_0$ , normalized, i.e.,  $x, y \sim N(0, 1)$ . Then,

$$\begin{aligned} & \Pr(j \in A_1 \mid j \in A_0) \\ & \geq \Pr(x + y \geq 2\sqrt{\ln(n/k) - 0.5 \ln(2 \ln(n/k))} \mid x \geq \sqrt{2 \ln(n/k) - \ln(2 \ln(n/k))}) \\ & \geq \Pr(y \geq (2 - \sqrt{2})\sqrt{\ln(n/k) - 0.5 \ln(2 \ln(n/k))}) \\ & \geq \mu_0 = \left(\frac{k}{n}\right)^{-(\sqrt{2}-1)^2}. \end{aligned}$$

## 57:12 Computation with Assemblies

For general  $t > 1$ , let  $x$  be the signal from the stimulus  $y$  from the overlap  $A_t \cap A_{t-1}$  and  $z$  from the rest of  $A_t$ . Then, with  $z \sim N(0, (1 - \mu_t))$ ,

$$\begin{aligned}
\mu_{t+1} &= \Pr(j \in A_{t+1} \mid j \in A_t) \\
&\geq \Pr(x + y + z \geq 2\sqrt{\ln(n/k) - 0.5 \ln(2 \ln(n/k))} \mid x \geq \sqrt{2 \ln(n/k) - \ln(2 \ln(n/k))}, \\
&\quad \text{and } y \geq \mu_t(2 - \sqrt{2})\sqrt{\ln(n/k) - 0.5 \ln(2 \ln(n/k))}) \\
&\geq \Pr(x \geq (2 - \sqrt{2})(1 - \mu_t)\sqrt{\ln(n/k) - 0.5 \ln(2 \ln(n/k))}) \\
&\geq \left(\frac{k}{n}\right)^{-(\sqrt{2}-1)^2(1-\mu_t)} \\
&= \mu_0^{1-\mu_t}.
\end{aligned}$$

The probability that a neuron  $j$ , which enters the cap at the first step, stays in the cap is thus at least

$$\begin{aligned}
\prod_t \mu_t &\geq \mu_0 \cdot \mu_0^{1-\mu_0} \cdot \mu_0^{1-\mu_0^{1-\mu_0}} \cdot \dots \\
&= \mu_0^{1+(1-\mu_0)+(1-\mu_0^{1-\mu_0})+\dots} \\
&\geq \mu_0^{1+(1-\mu_0)+(1-\mu_0)^2+(1-\mu_0)^3+\dots} \\
&= \mu_0^{\frac{1}{\mu_0}}
\end{aligned}$$

where we used the fact that  $1 - \mu_0^{(1-\mu_0)^i} = 1 - (1 - (1 - \mu_0))^{(1-\mu_0)^i} \geq (1 - \mu_0)^{i+1}$ .

So far, the computation was only for neurons that were in the very first caps. For neurons that make their first entrance later, the calculation is a bit different. Suppose a neuron enters the cap for the first time at iteration  $t$ . For general  $t > 1$ , let  $x$  be the signal from the stimulus  $y$  from the overlap  $A_t \cap A_{t-1}$  and  $z$  from the rest of  $A_t$ . Then, with  $z \sim N(0, (1 - \mu_t))$ , noting that  $x, y$  make up  $(1 + \mu_t)/2$  of the threshold  $C_t$ ,

$$\begin{aligned}
\mu_{t+1} &= \Pr(j \in A_{t+1} \mid j \in A_t) \\
&\geq \Pr(x + y + z \geq 2\sqrt{\ln(n/k) - 0.5 \ln(2 \ln(n/k))} \mid x + y \\
&\geq (1 + \mu_t)\sqrt{\ln(n/k) - \ln(2 \ln(n/k))}) \\
&\geq \Pr(x \geq (1 - \mu_t)\sqrt{\ln(n/k) - 0.5 \ln(2 \ln(n/k))}) \\
&\geq \left(\frac{k}{n}\right)^{-(1-\mu_t)/2} \\
&= \mu^{1-\mu_t}.
\end{aligned}$$

Note that  $\mu$  here is smaller than  $\mu_0$  for neurons that enter in the first cap. The computation for later steps, for such a neuron is similar, and we get that the probability that such a neuron stays in the cap forever is

$$\prod_t \mu_t \geq \mu \cdot \mu^{1-\mu} \cdot \mu^{1-\mu^{1-\mu}} \cdot \dots \geq \mu^{\frac{1}{\mu}}$$

as before. This completes the first part for  $\beta = 0$ .

For the second part, with  $\beta > 0$ , the calculation follows the same outline, except that the signal from the input is boosted by a factor of  $(1 + \beta)$  in each iteration, and the signal from previous caps is boosted by  $(1 + \beta)$  for a diminishing fraction  $\prod_t \mu_t$ . Ignoring the latter boost (for a lower bound),

$$\begin{aligned} \mu_{t+1} &\geq \Pr(x + y + z \geq 2\sqrt{\ln(n/k) - 0.5\ln(2\ln(n/k))} \mid x \geq \sqrt{2\ln(n/k) - \ln(2\ln(n/k))}, \\ &\quad \text{and } y \geq \mu_t(2 - \sqrt{2})\sqrt{\ln(n/k) - 0.5\ln(2\ln(n/k))}) \\ &\geq \Pr(x \geq (2 - \sqrt{2})(1 + \beta)^t(1 - \mu_t)\sqrt{\ln(n/k) - 0.5\ln(2\ln(n/k))}) \\ &\geq \left(\frac{k}{n}\right)^{-(\sqrt{2} - (1 + \beta)^t)^2(1 - \mu_t)} \\ &= \mu^{(1 - t\beta)(1 - \mu_t)}. \end{aligned}$$

We can now lower bound the probability of a neuron staying in the cap once it enters, and thereby the expected size of the total support. ◀

**Locality Sensitivity of Assemblies.** Returning to the motivating story on fly olfaction, is the assembly projection operation as locality sensitive as the simpler variant in insects? It appears that overlap of assemblies is an important indication of affinity of various sorts (co-occurrence, correlation, connection, similarity, etc.), and thus it matters whether or not it is preserved in projection. What we are able to show is that, if two sets of  $k$  cells overlap in a fraction of  $\alpha$ , and these two sets are projected sequentially to the same brain area, *the cores* of two resulting assemblies will share at least  $\lambda^2$  fraction of the overlap of their initial projections (given by Theorem 3); recall that  $\lambda$  is the size of the core over  $k$ , and for the parameters of interest is about half. Such a modest overlap at the core – the best connected part of the assembly – is a good omen for a large overlap of the two assemblies that will eventually emerge, an intuition that is supported by simulations, see Figure 1.<sup>7</sup>

## 5 Computing with Assemblies

The assembly hypothesis proposes that assemblies are the standard representations used in higher brain functions – memory, language, reasoning, decision-making, planning, math, music, story-telling and discourse – suggesting a grand and mysterious computational system with assemblies at its center, its basic data type. *How does this computational system work?* Foremost, what are its elementary operations?

- Assemblies do appear to *project* (see the discussion in [12] for an inspiring description of the process in the mouse piriform cortex): this is about the only way that assemblies can be created, and projection appears to be a most useful operation – in fact, in its absence, it is hard to imagine what assemblies may be good for. We denote the operation of an assembly  $x$  projecting to area  $A$  to create a new assembly  $y$  as  $\mathbf{project}(x, A, y)$  (the area of assembly  $x$ , denoted  $\mathbf{area}(x) \neq A$ , is implicit). Henceforth,  $\mathbf{parent}(y) = x$ <sup>8</sup>. Through  $\mathbf{project}$ , arbitrary relations can be maintained, with brain areas being the columns and time steps the rows; for example, a recent experiment [11] seems to suggest that the “subject-verb-object” relation in natural language may be achieved this way.

<sup>7</sup> We can prove something weaker, namely that substantial overlap persists to the assemblies, albeit only for sufficiently high plasticity, and under the additional assumption that the synaptic weights from the first projection have “faded” enough by homeostasis.

<sup>8</sup> As we shall see, some operations such as  $\mathbf{reciprocal-project}$  make the  $\mathbf{parent}$  function ambiguous, but we shall be ignoring this issue here.

- We also know from experiments [15, 9] that assemblies *associate* by exchanging cells (apparently a few percentage points of their support) when they become related through co-occurrence in the world and perhaps through other acquired relations. We denote this by  $\text{associate}(x, y)$  –  $x$  and  $y$  should of course be in the same area. It can be provably carried out by activating  $\text{parent}(x)$  and  $\text{parent}(y)$ , assumed to be in different areas, for a few steps [17]. It is natural to hypothesize that cell sharing between  $x$  and  $y$  has the effect that  $y$  may be henceforth activated, with some non-zero probability, when  $x$  is activated, and vice-versa. This opens up intriguing possibilities of sophisticated probabilistic reasoning and programming, and we suspect that much of the power of the assembly model may lie in this direction – which however we do not explore or exploit here.
- On another front, recent fascinating experiments [10, 24, 25, 16] suggest that *language processing* in humans involves the building and maintenance of syntactic structures such as syntax trees, and it is natural to assume that assemblies representing words are implicated there as well. We postulate the operation  $\text{merge}(x, y, A, z)$  which takes two assemblies  $x, y$  in different areas, and projects them *both* to assembly  $z$  in a third area  $A$ . Merge, the ability to consider two things as one, has been hypothesized in linguistics to be the quintessence of syntax, see for example [4]. It follows from the results in this paper that it can be implemented in our framework.
- A more complex and very useful operation is  $\text{reciprocal-project}(x, A, y, B, z)$  which creates in two areas  $A$  and  $B$  two assemblies  $y$  and  $z$  that can activate one another (while  $y$  can be activated by  $x$ , as in ordinary  $\text{project}$ ). It is assumed that there is synaptic connectivity from  $\text{area}(x)$  to  $A$  and both ways between  $A$  and  $B$ . The original assembly  $x$ , residing in a third area, can activate directly  $y$ . We conjecture that this operation can be carried out in our framework with high probability; it works reliably in simulations.  $\text{reciprocal-merge}$  is a straightforward generalization, which seems useful for language generation. Finally, another related operation is  $\text{append}(x, A, y)$ , useful for creating sequences, which we do not detail here.

## 5.1 The Power of Computation with Assemblies

According to the assembly hypothesis, assemblies and their operations are crucial for higher mental activities such as planning, language, and reason. The question may then arise: Is this purported computational system powerful enough? In particular, *is it Turing complete?* Many computer scientists are by instinct dubious about the value of such a pursuit; we agree, and in addition we are convinced that, if the assembly hypothesis is correct, the computational power of assemblies is wielded through means that are orthogonal to computer programming. On the other hand, an assessment of the computational power of this system can usefully inform our modeling, and in particular our search for essential primitives.

To continue on this path, we must create a programming system, formal enough to address the Turing completeness question, for writing simple programs with lines such as

```
if  $\text{area}(y) = A$ ,  $\text{project}(\text{parent}(y), B, z)$ .
```

To this end, we need to assume an environment in which names of assemblies, once declared – typically in a command such as  $\text{project}(x, A, y)$  – can be used in subsequent steps of the same program (area names are finite and fixed). Also, we introduce certain new primitives:  $\text{activate}(x)$  simply activates assembly  $x$  for a few steps; that is, we assume that  $\text{project}$  creates as a side-effect a *fuse* that can activate the new assembly. Also, we assume that



the downstream synapses from area  $A$  to area  $B$  are by default inactive, and must be activated explicitly by the operation `enable( $A, B$ )`. To illustrate, `project( $x, A, y$ )` is almost equivalent to

```
enable(area( $x$ ),  $A$ ); repeat  $T$  times: activate( $x$ ); disable(area( $x$ ),  $A$ ),
```

missing only a mechanism that names the new assembly  $y$ . Here  $T$  is the number of spikes required for assembly projection (about a dozen in simulations). Of course, it is debatable how realistically one expect such a programming framework to be operating in the brain.

We also introduce a *read* operation<sup>9</sup> returning information about the assemblies that are presently active, and their areas. Notice that all this assumes a simple computational mechanism acting as an *interpreter*, and lying outside our framework<sup>10</sup>.

Finally, we must address the issue of *reliability* in assembly computation. We shall make some assumptions:

- Any newly created assembly is a *random* set of  $k = \gamma\sqrt{n}$  neurons in its area.
- Two assemblies can interfere destructively in their operations, for example by spurious associations between them, but only if they overlap in more than  $\epsilon\sqrt{n}$  cells; the literature seems to suggest that  $\epsilon$  is at least 1%.
- At last we need to introduce *homeostasis*:. We assume that synaptic weights *fade* with time, regressing to the value 1. That is, at every time step weight  $w$  becomes  $\max\{\frac{w}{(1+\beta')}, 1\}$ , where  $0 < \beta' \ll \beta$ , the plasticity parameter.<sup>11</sup>  
Fading is both realistic and necessary for the simulation, since in its absence the computational system cannot erase information, and is therefore severely limited.
- Fading means that eventually all assemblies will lose their synaptic density and connection with their parent. To prevent this, we introduce *permanent versions* of operations such as `project`. For example, `permanent_project( $x, A, y$ )` involves, besides executing an ordinary `project` operation, repeating `activate( $x$ )` every  $\tau$  steps (with synaptic connections between the two areas in focus enables), where  $\tau$  is a small constant, much smaller than  $\frac{\beta}{\beta'}$ , either indefinitely or until an explicit `fade( $y$ )` command. There is evidence that such processes do happen in the brain, for example by fading, or reviving through rehearsal raw memory traces in the hippocampus.

The following is needed in the proof of the main result:

► **Lemma 6.** *The probability that a new assembly will interact destructively with a particular already existing assembly in the same area is at most  $\exp(-\frac{\epsilon\sqrt{n}}{\gamma^2})$ .*

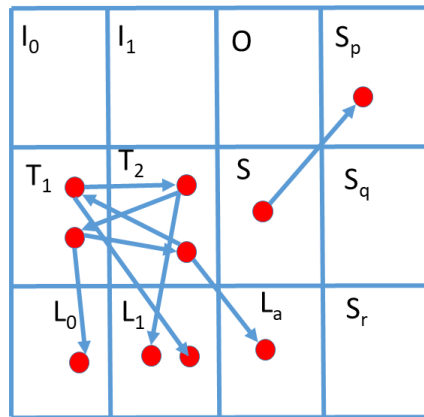
► **Theorem 7.** *The computational system described above can correctly simulate arbitrary  $O(\sqrt{n})$ -space computations with probability  $1 - \exp(O(\sqrt{n}))$ .*

**Sketch:** A Turing machine with a one-way circular tape of length  $m = O(\sqrt{n})$ , tape alphabet  $\Sigma$  and state set  $K$  can be simulated by a program of assembly operations. Let us assume the input-output convention that a new assembly appears in one of two designated input areas  $I_0, I_1$  at designated and well separated times, encoding a binary input tape; and that, upon

<sup>9</sup> Following a suggestion by Buszáki [5] that assemblies must be accompanied by a reader mechanism – as Buszáki puts it: “if a tree falls in the forest and there is nobody around to hear it fall, has it really fallen?”

<sup>10</sup> We do realize this is a strong assumption, unlikely to be literally true; we expect that the computational power of assemblies is realized through more organic means

<sup>11</sup> An equivalent, and perhaps more realistic, model of homeostasis would be to normalize the incoming weights of each neuron separately.



■ **Figure 3** Our representation of configuration (state, circular tape contents)  $[p, 011a]$ .

accepting termination, an assembly will appear in another area  $T$ . The Turing machine will be simulated by  $|\Sigma| + |K| + 6$  brain areas: the three input-output areas  $I_1, I_0, O$ , two areas for representing the tape denoted  $T_1$  and  $T_2$ , one area for representing the current state, denoted  $S$ , plus one area for each tape symbol  $a$  and state  $q$ , denoted, respectively,  $L_a$  and  $S_q$ . See Figure 3.

In the input phase, while the input is read from either  $I_0$  or  $I_1$  (depending on whether the input symbol is 0 or 1, assumed both to be in  $\Sigma$  (recall the input-output conventions), a chain of assemblies is created projecting back and forth between the two  $T_i$  areas (see Figure) through *permanent project* operations.

Each assembly in these two areas represents a tape square. The current symbol  $a$  in this square is represented through a projection to an assembly in area  $L_a$ , a projection that is permanent until it is explicitly faded when the same tape square is scanned again.

Similarly, another standard assembly  $s$  in area  $S$  points, through a projection (non-permanent, since the state changes at every step), to an area  $S_q$  representing the current state  $q$  (initially the starting state). The synapses from  $S$  to  $S_q$  are enabled, while the synapses from  $S$  to all other  $S_p$ 's are not<sup>12</sup>.

When the square corresponding to an assembly  $x$ , in one of the areas  $T_1, T_2$ , is scanned by the tape head, then  $x$  and  $s$  fire and a **read** is issued. Depending on the areas where assembly activity is read, say  $S_q$  and  $L_a$ , the correct current symbol  $a$  and state  $q$  are identified. Suppose that Turing machine's transition is  $\delta(q, a) = (p, b)$ . The synapses from  $S$  to  $S_q$  are disabled and those to  $S_p$  enabled, the assembly representing the previous symbol  $q$  is faded, and *permanent\_project*( $x, L_b, y$ ) is executed to record the current symbol of the tape square represented by  $x$ ; similarly for state. Then  $x$  fires again and a read is issued, to identify the tape assembly corresponding to the tape square that is next, and the computation continues. The straightforward details are omitted. ◀

<sup>12</sup>Notice that this effectively stores the state in the current instruction of the program; it can be done in more natural ways.

## 6 Discussion and open questions

We have identified a basic computational operation – random synaptic projection to a brain area followed by the selection, through inhibition, of the  $k$  neurons with the highest synaptic input – that appears to be ubiquitous in the animal brain and also useful for implementing more complex operations, but also happens to be mathematically concrete, productive, and interesting. Assembly projection can be the basis of a computational system at an intermediate level of abstraction – and unlike anything else that we have seen in theoretical neuroscience. Such a system, we hypothesize, may underlie the higher mental functions of the human brain – not an intensely researched subject in neuroscience. This hypothesis must be pursued both analytically, and – importantly – experimentally. We also believe that this line of work, and the rather simple and concrete model of brain operation it entails involving distinct brain areas, random graph connections, inhibition through cap, and probabilistic analysis, may constitute a promising entry point for theoretical computer scientists who want to work on brain-related problems. One of the contributions of this paper is pointing out the locality sensitive nature of assembly projection; this, together with the computational nature of association (which we did not consider here) promise to be important future directions for this work.

Assemblies may be implicated in implementing *natural language* in the human brain. Many recent experimental papers, see [24, 25, 11, 16, 10] among many others, appear to suggest that assembly-like operations like **projection** and **merge** may be implicated in language generation and processing.

We conclude with some more precise questions, that are motivated directly by our findings, and will help solidify the mathematical theory of assemblies, some of which we have already discussed in context in this paper.

1. Assembly support size. Is there a phase transition in the support size of an assembly (from  $\omega(k)$  to  $k + o(k)$ ) as the plasticity parameter  $\beta$  increases?
2. Assembly convergence. For high plasticity and with high probability, the limit of the random project plus cap process is a single fixed subset of size  $k$ . What are other possible limiting behaviors? E.g., is it possible to get two subsets of size  $k$  (possibly overlapping) that fire alternately? (We know cases where this happens at a small scale, that is, the two subsets of size  $k$  differ in 1-3 cells.) Will the limit have a common core (of what size as a function of plasticity) that always fires? Is the limit an activity pattern of finite length/description?
3. Model. Can our results be extended to less stylized models in which neurons fire asynchronously, or there is explicit inhibition (instead of cap)?
4. Base graph. We have assumed the base graph to have independently chosen edges. What is a deterministic condition on the base graph that suffices? E.g., is it enough to have expansion and roughly uniform degrees? Is global expansion necessary or do sufficiently strong local properties suffice (e.g., degree and co-degree)?
5. Extending GNP. Are richer models, e.g., those with higher reciprocity or triangle density, useful? For example, do they enable more powerful or efficient computations?
6. Computational power. Show that randomized  $s(n)$  space bounded computation can be simulated with  $n$  neurons and  $O(1)$  brain areas for some function  $s(n)$  larger than  $\sqrt{n}$ .
7. Capacity. Suppose that, in a brain area, we want to maintain with high probability pairwise intersections: two assemblies that intersect in a large ( $\alpha$  or more, say) fraction of their support should continue to so intersect, and similarly for pairs that intersect in less than  $\alpha$  fraction. For how many assemblies can we guarantee this invariant, as a function of  $n$ ?

8. Learning. Can assemblies perform learning (supervised or unsupervised)? Simulations suggest that assemblies can learn well-separated half-spaces quite naturally. Can this be proved formally? And what more ambitious forms of learning through assemblies are possible?
9. Assemblies vs 1-step Projections. Are assemblies (created as the limit of iterated random-project-and-cap) better for learning than 1-step (insect-like) projections? Is the recurrence of the mammalian brain a bonus or a handicap for learning?
10. Articulate a brain architecture for syntax (the building of syntactic trees) based on the assemblies operations project and merge and involving the medial temporal lobe, the superior temporal gyrus, and Broca's area of the left human brain.

---

### References

- 1 Rosa I. Arriaga, David Rutter, Maya Cakmak, and Santosh S. Vempala. Visual Categorization with Random Projection. *Neural Computation*, 27(10):2132–2147, 2015. doi:10.1162/NECO\_a\_00769.
- 2 Rosa I. Arriaga and Santosh Vempala. An algorithmic theory of learning: Robust concepts and random projection. *Machine Learning*, 63(2):161–182, 2006. doi:10.1007/s10994-006-6265-7.
- 3 M.F. Balcan, A. Blum, and S. Vempala. Kernels as Features: On Kernels, Margins, and Low-dimensional Mappings. *Machine Learning*, 65(1):79–94, 2006.
- 4 Robert C Berwick and Noam Chomsky. *Why only us: Language and evolution*. MIT Press, 2016.
- 5 G Buzsaki. Neural syntax: cell assemblies, synapsembles, and readers. *Neuron*, 68(3), 2010.
- 6 Sophie JC Caron, Vanessa Ruta, LF Abbott, and Richard Axel. Random convergence of olfactory inputs in the drosophila mushroom body. *Nature*, 497(7447):113, 2013.
- 7 S. DasGupta. Learning mixtures of Gaussians. In *Proc. of FOCS*, 1999.
- 8 Sanjoy Dasgupta, Charles F. Stevens, and Saket Navlakha. A neural algorithm for a fundamental computing problem. *Science*, 358(6364):793–796, 2017. doi:10.1126/science.aam9868.
- 9 Emanuela De Falco, Matias J Ison, Itzhak Fried, and Rodrigo Quian Quiroga. Long-term coding of personal and universal associations underlying the memory web in the human brain. *Nature Communications*, 7:13408, 2016.
- 10 Nai Ding, Lucia Melloni, Hang Zhang, Xing Tian, and David Poeppel. Cortical tracking of hierarchical linguistic structures in connected speech. *Nature neuroscience*, 19(1):158, 2016.
- 11 S. M. Frankland and J. D. Greene. An architecture for encoding sentence meaning in left mid-superior temporal cortex. *Proceedings of the National Academy of Sciences*, 112(37):11732–11737, 2015.
- 12 Kevin M Franks, Marco J Russo, Dara L Sosulski, Abigail A Mulligan, Steven A Siegelbaum, and Richard Axel. Recurrent circuitry dynamically shapes the activation of piriform cortex. *Neuron*, 72(1):49–56, 2011.
- 13 Kenneth D Harris. Neural signatures of cell assembly organization. *Nature Reviews Neuroscience*, 6(5):399, 2005.
- 14 Kenneth D Harris, Jozsef Csicsvari, Hajime Hirase, George Dragoi, and György Buzsáki. Organization of cell assemblies in the hippocampus. *Nature*, 424(6948):552, 2003.
- 15 Matias J Ison, Rodrigo Quian Quiroga, and Itzhak Fried. Rapid encoding of new memories by individual neurons in the human brain. *Neuron*, 87(1):220–230, 2015.
- 16 Sheena A Josselyn, Stefan Köhler, and Paul W Frankland. Finding the engram. *Nature Reviews Neuroscience*, 16(9):521, 2015.

- 17 R. Legenstein, W. Maass, C. H. Papadimitriou, and S. S. Vempala. Long-term Memory and the Densest k-subgraph Problem. In *Proc. of 9th Innovations in Theoretical Computer Science (ITCS) conference, Cambridge, USA, Jan 11-14. 2018*, 2018.
- 18 Robert Legenstein, Christos H Papadimitriou, Santosh Vempala, and Wolfgang Maass. Assembly pointers for variable binding in networks of spiking neurons. *arXiv preprint*, 2016. [arXiv:1611.03698](https://arxiv.org/abs/1611.03698).
- 19 Eva Pastalkova, Vladimir Itskov, Asohan Amarasingham, and György Buzsáki. Internally generated cell assembly sequences in the rat hippocampus. *Science*, 321(5894):1322–1327, 2008.
- 20 C. Pokorný, M. J. Ison, A. Rao, R. Legenstein, C. Papadimitriou, and W. Maass. Associations between memory traces emerge in a generic neural circuit model through STDP. *bioRxiv:188938*, 2017.
- 21 Leslie G. Valiant. *Circuits of the mind*. Oxford University Press, 1994.
- 22 Leslie G. Valiant. A neuroidal architecture for cognitive computation. *J. ACM*, 47(5):854–882, 2000. doi:10.1145/355483.355486.
- 23 Santosh Srinivas Vempala. *The Random Projection Method*, volume 65 of *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*. DIMACS/AMS, 2004. URL: <http://dimacs.rutgers.edu/Volumes/Vol165.html>.
- 24 Emiliano Zaccarella and Angela D. Friederici. Merge in the Human Brain: A Sub-Region Based Functional Investigation in the Left Pars Opercularis. *Frontiers in Psychology*, 6:1818, 2015. doi:10.3389/fpsyg.2015.01818.
- 25 Emiliano Zaccarella, Lars Meyer, Michiru Makuuchi, and Angela D Friederici. Building by syntax: the neural basis of minimal linguistic structures. *Cerebral Cortex*, 27(1):411–421, 2017.



# Local Computation Algorithms for Spanners

Merav Parter<sup>1</sup>

Weizmann IS, Rehovot, Israel  
merav.parter@weizmann.ac.il

Ronitt Rubinfeld<sup>2</sup>

CSAIL, MIT, Cambridge, MA, USA and TAU, Tel Aviv, Israel  
ronitt@csail.mit.edu

Ali Vakilian<sup>3</sup>

CSAIL, MIT, Cambridge, MA, USA  
vakilian@mit.edu

Anak Yodpinyanee<sup>4</sup>

CSAIL, MIT, Cambridge, MA, USA  
anak@mit.edu

---

## Abstract

---

A graph spanner is a fundamental graph structure that faithfully preserves the pairwise distances in the input graph up to a small multiplicative stretch. The common objective in the computation of spanners is to achieve the best-known existential size-stretch trade-off *efficiently*.

Classical models and algorithmic analysis of graph spanners essentially assume that the algorithm can read the input graph, construct the desired spanner, and write the answer to the output tape. However, when considering massive graphs containing millions or even billions of nodes not only the input graph, but also the output spanner might be too large for a single processor to store.

To tackle this challenge, we initiate the study of *local computation algorithms (LCAs)* for graph spanners in general graphs, where the algorithm should locally decide whether a given edge  $(u, v) \in E$  belongs to the output (sparse) spanner or not. Such LCAs give the user the “illusion” that a *specific* sparse spanner for the graph is maintained, without ever fully computing it. We present several results for this setting, including:

- For general  $n$ -vertex graphs and for parameter  $r \in \{2, 3\}$ , there exists an LCA for  $(2r - 1)$ -spanners with  $\tilde{O}(n^{1+1/r})$  edges and sublinear probe complexity of  $\tilde{O}(n^{1-1/2r})$ . These size/stretch trade-offs are best possible (up to polylogarithmic factors).
- For every  $k \geq 1$  and  $n$ -vertex graph with maximum degree  $\Delta$ , there exists an LCA for  $O(k^2)$  spanners with  $\tilde{O}(n^{1+1/k})$  edges, probe complexity of  $\tilde{O}(\Delta^4 n^{2/3})$ , and random seed of size  $\text{polylog}(n)$ . This improves upon, and extends the work of [Lenzen-Levi, ICALP’18].

We also complement these constructions by providing a polynomial lower bound on the probe complexity of LCAs for graph spanners that holds even for the simpler task of computing a sparse connected subgraph with  $o(m)$  edges.

To the best of our knowledge, our results on 3 and 5-spanners are the first LCAs with sublinear (in  $\Delta$ ) probe-complexity for  $\Delta = n^{\Omega(1)}$ .

**2012 ACM Subject Classification** Theory of computation  $\rightarrow$  Sparsification and spanners, Theory of computation  $\rightarrow$  Sketching and sampling

---

<sup>1</sup> MP is supported by Minerva Foundation (124042) and ISF-2084/18.

<sup>2</sup> RR is supported by the NSF grants CCF-1650733, CCF-1733808, IIS-1741137 and CCF-1740751.

<sup>3</sup> AV is supported by the NSF grant CCF-1535851.

<sup>4</sup> AY is supported by the NSF grants CCF-1650733, CCF-1733808, IIS-1741137 and the DPST scholarship, Royal Thai Government.





**Keywords and phrases** Local Computation Algorithms, Sub-linear Algorithms, Graph Spanners

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2019.58

## 1 Introduction

One of the fundamental structural problems in graph theory is to find a *sparse* structure which preserves the pairwise distances of vertices. In many applications, it is crucial for the sparse structure to be a *subgraph* of the input graph; this problem is called the *spanner* problem. For an input graph  $G = (V, E)$ , a  $k$ -spanner  $H \subseteq G$  (for  $k \geq 1$ ) satisfies that for any  $v, u \in V$ , the distance from  $v$  to  $u$  in  $H$  is at most  $k$  times the distance from  $v$  to  $u$  in  $G$ , where  $k$  is referred to as the *stretch* of the spanner. Furthermore, to reduce the *cost* of the solution, it is desired to output a minimum size/weight such subgraph  $H$ . The notion of spanners was introduced by Peleg and Schäffer [31] and has been used widely in different applications such as routing schemes [3, 30], synchronizers [2, 32], SDD's [37] and spectral sparsifiers [19].

It is folklore that for every  $n$ -vertex graph  $G$ , there exists a  $(2k - 1)$ -spanner  $H \subseteq G$  with  $O(n^{1+1/k})$  edges. In particular, if the *girth conjecture* of Erdős [14] is true, then this size-stretch trade-off is optimal. Spanners have been considered in many different models such as distributed algorithms [5, 8–11, 13, 33] and dynamic algorithms [4, 6, 12].

**Local computation of small stretch spanners.** When the graph is so large that it does not fit into the main memory, the existing algorithms are not sufficient for computing a spanner. Instead, we aim at designing an algorithm that answers *queries* of the form “is the edge  $(u, v)$  in the spanner?” without computing the whole solution upfront. One way to get around this issue is to consider the *Local Computation Algorithms (LCAs)* model (also known as the *Centralized Local model*), introduced by Rubinfeld et al. [35] and Alon et al. [1]. There can be many different plausible  $k$ -spanners; however, the goal of LCAs for the  $k$ -spanner problem is to design an algorithm that, given access to *primitive* probes (i.e. NEIGHBOR, DEGREE and ADJACENCY probes) on the input graph  $G$ , for each *query* on an edge  $e \in E(G)$  *consistently* with respect to a *unique*  $k$ -spanner  $H \subseteq G$  (picked by the LCA arbitrarily), outputs whether  $e \in H$ . The performance of the LCA is measured based on the *quality of solution* (i.e. number of edges in  $H$ ) and the *probe complexity* (the maximum number of probes per each query) of the algorithm<sup>5</sup>. In other words, an LCA gives us the “illusion” as if we have query access to a precomputed  $k$ -spanner of  $G$ .

The study of LCAs with *sublinear* probe complexity for *nearly linear* size spanning subgraphs (or *sparsifiers*) is initiated by Levi, Ron, and Rubinfeld [24, 25] for some restricted families of graphs such as minor-closed families. However, their focus is mainly on designing LCAs that preserve the *connectivity* while allowing the stretch factor to be as large as  $n$ . Moreover, in their work, the input graph is sparse (has  $O(n)$  edges), while the classical  $k$ -spanner problem becomes relevant only when the input graph is dense (with superlinear number of edges). Recently, Lenzen and Levi [21] designed the first sparsifier LCA in *general graphs* with  $(1 + \varepsilon)n$  edges, stretch  $O(\log^2 n \cdot \text{poly}(\Delta/\varepsilon))$  and probe complexity of  $O(\text{poly}(\Delta/\varepsilon) \cdot n^{2/3})$ , where  $\Delta$  is the maximum degree of the input graph.

<sup>5</sup> We may also measure the *time complexity* of an LCA. In our LCAs, the time complexities are clearly only a factor of  $\text{poly}(\log n)$  higher than the corresponding probe complexities, so we focus our analysis on probe complexities.

In this work, we show that sublinear time LCAs for spanners are indeed possible in several cases. We give: (I) 3 and 5-spanners for general graphs with optimal trade-offs between the number of edges and the stretch parameter (up to polylogarithmic factors), and (II) general  $k$ -spanners, either in the dense regime (when the minimum degree is at least  $n^{1/2-1/(2k)}$ ) or in the sparse regime (when the maximum degree is  $n^{1/12-\varepsilon}$ ).

**Broader scope and agenda: local computation algorithms for dense graphs.** LCAs have been established by now for a large collection of problems, including Maximal Independent Set, Maximum Matching, and Vertex Cover [1, 15, 26, 27, 29, 34, 35]. These algorithms typically suffer from a probe complexity that is exponential in  $\Delta$  and thus are efficient only in the sparse regime when  $\Delta = O(1)$ .

To this end, obtaining LCAs even with a polynomial dependency in  $\Delta$  is a major open problem for many classical local graph problems, as noted in [16, 26, 28]. For instance, recently Ghaffari and Uitto [16] obtained an LCA for the MIS problem with probe complexity of  $\Delta^{O(\log \log \Delta)} \cdot \log n$  improving upon a long line of results. Their result also illustrates the connection between LCAs with good dependency on  $\Delta$ , and algorithms for the massively parallel computation model with sublinear space per machine. Recently, [26] and [21] provided LCAs with probe complexities *polynomial* in  $\Delta$  for the problems of  $(1-\varepsilon)$ -maximum matching and sparse connected subgraphs, respectively. Note that in the context of spanners, such algorithms are still inefficient when the maximum degree is polynomial in  $n$ , which is precisely the setting where graph sparsification is applied.

## 1.1 Our results and techniques

In this paper we initiate the study of LCAs for graph spanners in *general graphs* which concerns with the following task: *How can we decide quickly (e.g., sublinear in  $n$  time) if a given edge  $e$  belongs to a sparse spanner (with fixed stretch) of the input graph, without preprocessing and storing any auxiliary information?* In the design of LCAs for graph problems, the set of defined *probes* to the input graph plays an important role. Here we consider the following common probes: NEIGHBOR probes (“what is the  $i^{\text{th}}$  neighbor of  $u$ ?”), DEGREE probes (“what is  $\deg(u)$ ?”) and ADJACENCY probes (“are  $u$  and  $v$  neighbors?”) [17, 18]. We emphasize that the answer to an ADJACENCY probe on an ordered pair  $\langle u, v \rangle$  is the index of  $v$  in  $\Gamma(u)$  if<sup>6</sup> the edge exists and  $\perp$  otherwise. Note that if the maximum degree in the input graph is  $O(1)$ , each ADJACENCY probe can be implemented by  $O(1)$  number of NEIGHBOR probes.

The problem of designing LCAs for spanners is closely related to designing LCAs for *sparse connected subgraphs* with  $(1+\varepsilon)n$  edges which was first introduced by [24]. With the exception of [21], a long line of results for this problem usually concerns special *sparse* graph families, rather than general graphs. A summary of these results with a comparison to our results is provided in Table 1.

### 1.1.1 LCAs for 3 and 5-Spanners for General Graphs

Our first contribution is the local construction of 3 and 5-spanners for general graphs, while achieving the optimal trade-offs between the number of edges and the stretch factors (up to polylogarithmic factors)<sup>7</sup>. In particular, our LCAs have  $n^{1-\varepsilon}$  probe complexity even when the input graph is dense with  $\Delta = \Omega(n)$ ; note that in such a case, given a query

<sup>6</sup>  $\Gamma(u)$  denotes the neighbor set of  $u$ , whereas  $\Gamma^+(u) = \Gamma(u) \cup \{u\}$ .

<sup>7</sup> Indeed, the girth conjecture of Erdős is resolved for these stretch factors; see e.g., [39].

■ **Table 1** Table of results on LCAs for the spanner problem. The symbol ‘-’ indicates that the stretch is not analyzed. The input graph is a simple graph with  $n$  vertices,  $m$  edges, maximum degree  $\Delta$ , and belongs to the indicated graph family.  $\tilde{O}$  hides a factor of  $\text{poly}(\log n, k)$ .

Reference	Graph Family	# Edges	Stretch	Probe Complexity	
Prior works	[24]	Bounded Degree Graphs	$(1 + \varepsilon)n$	–	$\Omega(\sqrt{n})$
		Expanders	$(1 + \varepsilon)n$	–	$O(\sqrt{n})$
		Subexponential growth	$(1 + \varepsilon)n$	–	$O(\sqrt{n})$
	[23]	Minor-free	$(1 + \varepsilon)n$	$\text{poly}(\Delta, 1/\varepsilon)$	$\text{poly}(\Delta, 1/\varepsilon)$
	[25]	Minor-free	$(1 + \varepsilon)n$	$O((\log \Delta)/\varepsilon)$	$\text{poly}(\Delta, 1/\varepsilon)$
	[22]	Expansion $(1/\log n)^{1+o(1)}$	$(1 + \varepsilon)n$	super-exponential in $1/\varepsilon$	super-exponential in $1/\varepsilon$
[21]	General	$(1 + \varepsilon)n$	$O(\log^2 n \cdot \text{poly}(\Delta/\varepsilon))$	$O(n^{2/3} \cdot \text{poly}(\Delta/\varepsilon))$	
Here	Thm. 1	General	$\tilde{O}(n^{1+1/r})$	$2r - 1$ ( $r \in \{2, 3\}$ )	$\tilde{O}(n^{1-1/(2r)})$
	Thm. 12	Min degree $O(n^{1/2-1/(2k)})$	$\tilde{O}(n^{1+1/k})$	5	$\tilde{O}(n^{1-1/(2k)})$
	Thm. 2	Max degree $O(n^{1/12-\varepsilon})$	$\tilde{O}(n^{1+1/k})$	$O(k^2)$	$\tilde{O}(n^{1-4\varepsilon})$
	Thm. 3	General	$o(m)$	any $k \leq n$	$\Omega(\min\{\sqrt{n}, n^2/m\})$

edge  $(u, v)$ , the LCA should return yes or no without being able to inspect the neighbor lists  $\Gamma(u)$  and  $\Gamma(v)$ . In what follows we show how to manipulate the common distributed construction by Baswana and Sen [5] to yield LCAs for 3-spanners and 5-spanners with sublinear probe complexity.

**The common distributed approach.** Most distributed spanner constructions are based on thinning the graph via clustering: construct a random set  $S$  of *centers* by adding each vertex to  $S$  independently with some fixed probability. For each vertex  $v$  sufficiently close to a center in  $S$ , include the edges of the shortest path connecting  $v$  to its closest member  $s \in S$ : this induces a *cluster* around each center  $s \in S$ , where every pair of vertices in the same cluster are connected by a short path. Then, add edges connecting pairs of neighboring clusters to ensure the desired stretch factor.

The following algorithm constructs a 3-spanner  $H \subseteq G$  with  $\tilde{O}(n^{3/2})$  edges. First, add to  $H$  all edges incident to vertices of degree at most  $\sqrt{n}$ . Second, pick a collection  $S$  of centers by sampling each vertex independently with probability  $\Theta(\log n/\sqrt{n})$ . Each vertex  $v$  of degree at least  $\sqrt{n}$  picks a *single* neighboring center  $s \in S \cap \Gamma(v)$  (which exists w.h.p.) as its *center*, then adds  $(v, s)$  to  $H$ , forming a collection of  $|S| = O(\sqrt{n})$  clusters (stars) around these centers. Lastly, every vertex  $u$  adds only one edge to each of its neighboring clusters – note that this last step may add edges whose endpoints are both non-centers. This results in a 3-spanner: For omitted edge  $(u, v)$  in  $G$ , if  $u$  and  $v$  are in the same cluster, then they have a path of length 2 through their shared center  $s$ . If  $u$  and  $v$  are in different clusters, an edge from  $u$  to some other vertex  $w$  in  $v$ ’s cluster would have been chosen, providing the path  $\langle u, w, s, v \rangle$  of desired stretch 3 connecting  $u$  and  $v$ , where  $s$  is  $v$ ’s center.

**The challenge and key ideas.** Recall that our goal is to design an LCA for 3-spanners  $H \subseteq G$  of size  $\tilde{O}(n^{3/2})$  and probe complexity of  $\tilde{O}(n^{3/4})$ : the LCA is given an edge  $(u, v)$  and must answer whether  $(u, v) \in E(H)$ . First, if  $\text{deg}(u)$  or  $\text{deg}(v)$  is at most  $\sqrt{n}$ , then the algorithm can immediately say YES. This requires only two DEGREE probes for the endpoints  $u, v$ . Hence, the interesting case is where both  $u$  and  $v$  have degrees at least  $\sqrt{n}$ .

We start by sampling each vertex into the center set  $S$  with probability of  $p = \Theta(\log n/\sqrt{n})$ , thus w.h.p. guaranteeing that each high-degree vertex has at least one sampled neighbor. For clarity of explanation, assume that given the ID of a vertex  $v$ , the LCA algorithm can decide

(with no further probes) whether  $v$  is sampled. Upon selecting the set of centers  $S$ , the above mentioned distributed algorithm has two degrees of freedom (which our LCA algorithm will enjoy). First, for a high-degree vertex  $v$ , there could be potentially many sampled neighbors in  $S$ : the distributed algorithm lets  $v$  join the cluster of an arbitrarily sampled neighbor. The second degree of freedom is in connecting a high-degree vertex to neighboring clusters. In the distributed algorithm, a vertex connects to an arbitrarily chosen neighbor in each of its neighboring clusters. Since the answers of the LCA algorithm should be consistent, it is important to carefully fix these decisions to allow small probe complexity.

**The naïve approach for 3-spanners and its shortcoming.** The most naïve approach is as follows: for each  $v$ , traverse the list  $\Gamma(v)$  in a fixed order and pick the *first* neighbor that satisfies the required conditions. That is, a vertex joins the cluster of its *first* sampled neighbor (center) and connects to its *first* representative neighbor in each of its neighboring clusters. To analyze the probe complexity of such a construction, consider a query edge  $(u, v)$  where  $\deg(u), \deg(v) \geq \sqrt{n}$ . By probing for the first  $\sqrt{n}$  neighbors of  $u$  and  $v$ , one can compute the cluster centers  $c_u$  and  $c_v$  of  $u$  and  $v$  with high probability. The interesting case is where  $u$  and  $v$  belong to different clusters. In such a case, the LCA algorithm should say YES only if  $v$  is the first neighbor of  $u$  that belongs to the cluster of  $c_v$ . To check if this condition holds, the algorithm should probe for each of the neighbors  $w$  of  $u$  that appears before  $v$  in  $\Gamma(u)$ , and say NO if there exists such earlier neighbor  $w$  that belongs to the cluster of  $c_v$ . Here, it remains to show how this cluster-membership testing procedure is implemented.

A *cluster-membership test*, for a pair  $\langle s, w \rangle$  with  $s \in S$ , must return YES iff  $w$  belongs to the cluster of the center  $s$ . The above mentioned algorithm thus makes  $O(\deg(v))$  cluster-membership tests for each  $w$  preceding  $u$  in  $\Gamma(v)$  and  $s = c_v$ . Since each center is sampled with probability  $p = \log n / \sqrt{n}$ , the probe complexity of a single cluster-membership test is  $O(\sqrt{n})$  w.h.p., leading to a total probe complexity of  $O(\deg(v) \cdot \sqrt{n})$ .

**Idea (I) – Multiple centers for efficient cluster-membership test.** The key idea in our solution is to pick the cluster centers in a way that allows answering each cluster-membership test for a pair  $\langle s, w \rangle$  using a single NEIGHBOR probe! Towards this goal, we let each high-degree vertex join *multiple* clusters, instead of just one. In particular, for a vertex  $w$ , we look at the subset  $\Gamma_1(w)$  consisting of its first  $\sqrt{n}$  neighbors in  $\Gamma(w)$ . We then let  $w$  join the clusters of all sampled neighbors in  $\Gamma_1(w) \cap S$ . Since each vertex is a center with probability  $p$ , this implies that, w.h.p.,  $w$  joins  $\Theta(p \cdot |\Gamma_1(w)|) = \Theta(\log n)$  many clusters. Though this approach adds a multiplicative  $O(\log n)$  factor to the size of our spanner, it will pay off dramatically in terms of the probe complexity of our LCA. In particular, this modification enables the algorithm to test cluster-membership with a single ADJACENCY probe: the vertex  $w$  belongs to the cluster of  $s$ , if the index of  $s$  in  $w$ 's neighbor-list is at most  $\sqrt{n}$  (the index is returned by the ADJACENCY probe on  $u$  and  $s$ ). This idea alone decreases the probe complexity of our LCA to  $\tilde{O}(\deg(w))$ .

**Idea (II) – Neighborhood partitioning.** The multiple center technique above allows our LCA to handle edges adjacent to a vertex  $u$  of degree at most  $n^{3/4}$ . For  $\deg(u) > n^{3/4}$ , our LCA cannot afford to look at all neighbors of  $u$ . To this end, we *partition* the neighbors of  $u$  into *blocks* of size  $n^{3/4}$  each. Rather than adding only one edge between  $u$  to each neighboring cluster, we make the decision on which edges to keep for each block *independently*, by scanning only the block containing  $v$  and keeping  $(u, v)$  if  $v$  belongs to the cluster that was not previously seen in this block. Though this leads to an increase in the number of

edges by a factor of  $\deg(u)/n^{3/4} \leq n^{1/4}$ , we can now keep the probe complexity down to  $\tilde{O}(n^{3/4})$  as we only need to scan the block containing  $v$  given the query  $(u, v)$  instead of  $u$ 's entire neighbor-list. To keep the size of the spanner small, e.g.,  $\tilde{O}(n^{3/2})$ , we use the fact that  $O(n^{1/4} \log n)$  sampled vertices are enough to hit the neighborhoods of all vertices with degree more than  $n^{3/4}$  with high probability. Since for each block of size  $n^{3/4}$  in the neighborhood of  $u$  the algorithm adds  $O(|S|)$  edges, the total number of edges added per vertex is  $O(|S| \cdot \deg(u)/n^{3/4}) = \tilde{O}(n^{3/2})$ , as desired.

**Overview of the LCA for 5-spanners.** For 5-spanners, the desired number of edges is  $\tilde{O}(n^{4/3})$ . This allows us to immediately add to the spanner all edges incident to low-degree vertices  $u$  with  $\deg(u) = \tilde{O}(n^{1/3})$ . The common distributed construction for 5-spanners computes  $O(n^{2/3})$  clusters by sampling each center independently with probability  $\Theta(\log n/n^{1/3})$ . By letting each high-degree vertex (i.e., with  $\deg(u) = \Omega(n^{1/3})$ ) join the cluster of one of its sampled neighbors, the spanner contains a collection of  $O(n^{2/3})$  (vertex-disjoint) clusters that, w.h.p., cover all high-degree vertices. Finally, each pair of neighboring clusters  $C_1, C_2$  are connected by adding an edge  $(u, v) \in (C_1 \times C_2) \cap E$  to the spanner  $H$ . It is straightforward to verify that  $H$  is a 5-spanner of size  $\tilde{O}(n^{4/3})$ .

Designing LCAs for the 5-spanner problem turns out to be significantly more challenging than the 3-spanner case. The reason is that deciding whether an edge  $(u, v)$  is in the 5-spanner requires information from the *second neighborhoods* of  $v$  and  $u$ , which is quite cumbersome when one cannot even read the entire neighborhood of a vertex. Our solution extends the 3-spanner construction in two ways: some of the edges added to our 5-spanner are between *cluster* pairs, instead of edges between a vertex and a cluster as in the 3-spanner solution. Another set of edges added to the 5-spanner is between pairs of vertex and cluster, but unlike the 3-spanner case, these clusters have now *radius two*.

**Idea (III) – Cluster partitioning (bucketing).** The standard clustering-based construction of 5-spanners adds an edge between every pair of neighboring clusters (stars). This clustering-based construction cannot be readily implemented with the desired probe complexity. To see why, consider clusters centered at  $s$  and  $t$ , containing  $u$  and  $v$  respectively. A naïve attempt spends  $\deg(s) \cdot \deg(t)$  probes for vertices between these clusters, as to consistently pick a unique edge between the two clusters.

One of our tools extends the idea of neighborhood partitioning from 3-spanner into cluster partitioning. Each of the  $O(n^{2/3})$  clusters is partitioned into balanced *buckets* of size  $\Theta(n^{1/3})$ .<sup>8</sup> The algorithm then picks only one edge between any pair of neighboring buckets. Since the number of buckets can be shown to be  $\tilde{O}(n^{2/3})$ , the spanner size still remains  $\tilde{O}(n^{4/3})$ . Unlike partitioning *neighbor-lists*, partitioning a *cluster* requires the full knowledge of its members – which are no longer nicely indexed in a list. To be able to efficiently partition a clusters, the algorithm allows only vertices with degree at most  $n^{5/6}$  to be chosen as cluster centers. The benefit of this restriction is that one can inspect the entire neighborhood of a center in  $O(n^{5/6})$ . The drawback of this approach is that it only clusters vertices that have sufficiently many neighbors (i.e., at least  $n^{1/3}$ ) with degree less than  $n^{5/6}$ . The remaining vertices are handled via their high-degree neighbors (i.e., of degree at least  $n^{5/6}$ ) as described next.

<sup>8</sup> Note that each cluster may have at most one bucket of size  $o(n^{1/3})$ .

**Idea (IV) – Representatives.** Using the neighborhood-partitioning idea from 3-spanner, all vertices with degree at least  $n^{5/6}$  can be clustered by sampling  $\tilde{O}(n^{1/6})$  cluster centers. By partitioning the neighborhood of each high-degree vertex into disjoint blocks each of size  $\tilde{O}(n^{5/6})$ , one can construct a 3-spanner for all edges incident to these high-degree vertices with probe complexity of  $\tilde{O}(n^{5/6})$  while using  $\tilde{O}(n^{4/3})$  edges. To take care of vertices of degrees less than  $n^{5/6}$  that have many high-degree neighbors, we let them join the cluster of their high-degree neighbors, hence creating *clusters of depth 2*.

To choose which cluster to join (in the second level), our vertex, which has many high-degree neighbors, simply chooses and connects itself to one or more high-degree neighbors, called its *representatives*. To determine the representatives of a vertex  $u$ , we simply pick  $\Theta(\log n)$  random neighbors of  $u$ , and w.h.p. one of them will have high-degree, and hence is chosen as  $u$ 's representative.

We implement our LCA by first picking  $|S| = \tilde{O}(n^{1/6})$  centers. Consider the query edge  $(u, v)$  where  $\deg(u), \deg(v) \geq n^{1/3}$  and  $u$  has many high-degree neighbors. Here,  $u$  has  $\Theta(\log n)$  representatives, each of which has  $\Theta(\log n)$  centers in  $S$  w.h.p., so  $u$  belongs to  $O(\log^2 n)$  clusters. As in the 3-spanner case, we keep  $(u, v)$  if  $v$  is the first neighbor of  $u$  in the cluster that  $v$  belongs to. We find the representatives of each neighbor of  $u$  by making  $O(\log n)$  probes, and for all these  $\deg(u) \cdot O(\log n) = \tilde{O}(n^{5/6})$  representatives, check if they belong to any of  $v$ 's  $O(\log^2 n)$  clusters with  $\tilde{O}(n^{5/6})$  total probes.

► **Theorem 1** (3 and 5-spanners). *For every  $n$ -vertex simple undirected graph  $G$ , there exists an LCA for  $(2r - 1)$ -spanners with  $\tilde{O}(n^{1+1/r})$  edges and probe complexity  $\tilde{O}(n^{1-1/(2r)})$  for  $r \in \{2, 3\}$ . Moreover, the algorithm only uses a seed of  $O(\log^2 n)$  random bits.*

In fact, if  $G$  has *minimum* degree  $\omega(n^{1/3})$ , we may apply the 5-spanner construction (with modified parameters) to obtain 5-spanners with even smaller number of edges as indicated in Table 1 (Theorem 12): this minimum degree assumption indeed allows even sparser spanners, bypassing the girth conjecture that holds for *general* graphs. We also remark that, in the somewhat related setting of dynamic computation, spanner algorithms with worst-case *sublinear* update time are currently known only for 3 and 5-spanners as well (see Bodwin and Krinninger, [6]).

### 1.1.2 LCA for $O(k^2)$ -spanners

Our second contribution is the local construction of  $O(k^2)$ -spanners with  $O(n^{1+1/k})$  edges for any  $k \geq 1$ , which has sub-linear probe complexity for graphs of maximum degree  $\Delta = O(n^{1/12-\varepsilon})$ . Our approach improves upon and extends the recent work of Lenzen and Levi [21]. The work of [21] aims at locally constructing a spanning subgraph with  $O(n)$  edges, but the stretch parameter of their subgraph might be as large as  $O(\text{poly}(\Delta) \log^2 n)$ . In addition, this construction requires a random seed of polynomial size. In our construction, we reduce the stretch parameter of the constructed subgraph to  $O(k^2)$ , independent of both  $n$  and  $\Delta$ , while using only  $\tilde{O}(n^{1+1/k})$  edges. In addition, we implement our randomized constructions using  $\text{poly}(\log n)$  independent random bits, whereas [21] uses  $\text{poly}(n)$  bits. We remark that for the LCAs with large stretch parameter considered in [21], our techniques can still be applied to exponentially reduce the required amount of random bits, and save a factor of  $\Delta$  in the probe complexity. The details of  $O(k^2)$ -spanners are omitted in this version: please refer to the longer version of this paper for the missing details of this result.

► **Theorem 2** ( $O(k^2)$ -spanners). *For every integer  $k \geq 1$  and every  $n$ -vertex simple undirected graph  $G$  with maximum degree  $\Delta$ , there exists a (randomized) LCA for  $O(k^2)$ -spanner with  $\tilde{O}(n^{1+1/k})$  edges and probe complexity  $\tilde{O}(\Delta^4 n^{2/3})$ . Moreover, the algorithm only uses  $O(\log^2 n)$  random bits.*



The high level structure is as in [21]: for a given stretch parameter  $k$ , partition the edges in  $G$  into the *sparse* set  $E_{\text{sparse}}$  and the *dense* set  $E_{\text{dense}}$ . Roughly speaking, the sparse set  $E_{\text{sparse}}$  only consists of edges  $(u, v)$  for which the  $k$ -neighborhood in  $G$  of either  $u$  or  $v$  contains at most  $O(n^{2/3})$  vertices. For this sparse region in the graph, we can simulate a standard distributed algorithm for spanners [5, 7] (using only a poly-logarithmic number of random bits), with small probe complexity. This yields an LCA handling the sparse edges with  $O(\Delta^2 n^{2/3})$  probe complexity.

To take care of the dense edges, we sample a collection of  $O(n^{2/3} \log n)$  centers and partition the (dense) vertices into *Voronoi cells* around these centers.

The main challenge is in connecting the Voronoi cells, keeping in mind that taking an edge between every pair of cells adds too many edges to the spanner. To get around it, the main contribution of [21] was in designing a set of rules for connecting bounded-size sub-structures in Voronoi cells, called *clusters*. The high-level description of the rules are as follows<sup>9</sup>: *mark* a random subset of  $O(n^{1/3} \log n)$  Voronoi cells (among the  $n^{2/3}$  Voronoi cells), then *connect*<sup>10</sup> them according to the following rules using  $\tilde{O}(n)$  edges each. Rule (1): connect every marked Voronoi cells to each of its neighboring Voronoi cells. Rule (2): if a Voronoi cell has no neighboring marked Voronoi cells, then connect it to all its neighboring Voronoi cells as well. Rule (3): For each pair of (not necessarily adjacent) Voronoi cell  $\mathbf{a}$  and marked Voronoi cell  $\mathbf{c}$  sharing common neighboring Voronoi cells  $\Gamma(\mathbf{a}) \cap \Gamma(\mathbf{c})$ , keep an edge from  $\mathbf{a}$  to a single Voronoi cell  $\mathbf{b}^* \in \Gamma(\mathbf{a}) \cap \Gamma(\mathbf{c})$  (i.e.,  $\mathbf{b}^*$  has the minimum ID in  $\Gamma(\mathbf{a}) \cap \Gamma(\mathbf{c})$ ). This last rule handles the edges of (unmarked) Voronoi cells that have some neighboring marked Voronoi cell.

**Idea (V) – Establishing the  $O(k^2)$  stretch guarantee.** In our implementation, the radius of each Voronoi cell is  $O(k)$  (as opposed to  $O(\Delta \log n)$  in [21]). Thus, it suffices to show that the spanner path from Voronoi cell supervertices  $\mathbf{a}$  to  $\mathbf{b}$  only visits  $O(k)$  other Voronoi cells. To this end, we impose a random ordering of the Voronoi cells, by assigning them distinct random *ranks*. We then make the following modification to Rule (3): add an edge from  $\mathbf{a}$  to  $\mathbf{b}$  if there exists a marked Voronoi cell  $\mathbf{c}$  such that the rank  $r(\mathbf{b})$  of  $\mathbf{b}$  is among the  $O(n^{1/k} \log n)$  lowest ranks in  $\Gamma(\mathbf{a}) \cap \Gamma(\mathbf{c})$ , restricted to those discovered by the LCA. This modified rule allows us to extend the inductive connectivity argument of [21] to show that every pair of adjacent cells are connected by a path that goes through  $O(k)$  cells – since each cell has radius  $O(k)$ , the final stretch is  $O(k^2)$ .

**Idea (VI) – Graph connectivity with bounded independence.** One of our key technical contributions is in showing that one can implement the above randomized random rank assignment using small number of random bits. We show that the ranks of Voronoi cells can be computed using  $T = \Theta(k)$  hash functions  $h_1, \dots, h_T$  chosen uniformly at random form a family of  $O(\log n)$ -wise independent hash functions of the form  $\{0, 1\}^{\log n} \rightarrow \{0, 1\}^{O((\log n)/k)}$ . We define our rank function as a concatenation of  $h_i$ 's on the ID of the Voronoi cell's center: for the Voronoi cell centered at  $v$ ,  $r(v) = h_1(\text{ID}(v)) \circ \dots \circ h_T(\text{ID}(v))$ . We then carefully adopt the inductive stretch argument to this randomized rank assignment with limited independence so that in the  $i^{\text{th}}$  step, our analysis only relies on the hash function  $h_i$ .

<sup>9</sup> Here, we state a simplified version of the rules. In particular, the rules are expressed in terms of clusters whose exact definitions are skipped for now. Refer to the longer version of our paper for the precise definitions of the rules.

<sup>10</sup> We *connect* two vertex sets by adding the unique lexicographically-first edge between the two vertex sets (if any exists) based on the vertex IDs of the endpoints.



### 1.1.3 Lower Bounds

To establish the lower bound, we construct two distributions over undirected  $d$ -regular graph instances that contain a designated edge  $e$ . For graphs in the first family, it holds that after removing  $e$ , w.h.p., they remain connected while in the second family, removing  $e$  disconnects the endpoints of  $e$  and leave them in separate connected components. We show that for the edge  $e$ , any LCA that makes  $o(\min\{\sqrt{n}, n/d\}) = o(\min\{\sqrt{n}, n^2/m\})$  probes can only distinguish whether the underlying graph is from the first family or the second family with probability  $1/2 + o(1)$ .

Our approach mainly follows from the analysis of Kaufman, Krivelevich, and Ron [20], on the lower bound construction of [24]. While [20] studies a rather different problem of bipartiteness testing, we consider similar probe types and obtain a similar lower bound as those of [20]. On the other hand, the construction of [24] shows the probe complexity of  $\Omega(\sqrt{n})$  for LCAs for spanning graphs that only use NEIGHBOR probes, not ADJACENCY probes.

► **Theorem 3** (Lower Bound). *Any local randomized LCA that computes, with success probability at least  $2/3$ , a spanner of the simple undirected  $m$ -edge input graph  $G$  with  $o(m)$  edges, has probe complexity  $\Omega(\min\{\sqrt{n}, n^2/m\})$ .*

The details of this result are deferred to the longer version of this paper.

## 1.2 Model Definition and Preliminaries

**Graph notation.** Throughout, we consider simple unweighted undirected graphs  $G = (V, E)$  on  $n = |V|$  vertices and  $m = |E|$  edges. Each vertex  $v$  is labeled by a unique  $O(\log n)$ -bit value  $\text{ID}(v)$ <sup>11</sup>. For  $u \in V$ , let  $\Gamma(u, G) = \{v : (u, v) \in E\}$  be the neighbors of  $u$ ,  $\deg(u, G) = |\Gamma(u, G)|$  be its degree, and define  $\Gamma^+(u, G) = \Gamma(u, G) \cup \{u\}$ . Denote  $V_I = \{v \in V : \deg(v, G) \in I\}$  where  $I$  is an interval. For  $u, v \in V$ , let  $\text{dist}(u, v, G)$  be the shortest-path distance between  $u$  and  $v$  in  $G$ . Let  $\Gamma^k(u, G) = \{v : \text{dist}(u, v, G) \leq k\}$  be the  $k^{\text{th}}$ -neighborhood of  $u$ , and denote its size  $\deg_k(u, G) = |\Gamma^k(u, G)|$ . For subsets  $V_1, V_2 \subseteq V$ , let  $E(V_1, V_2) = E \cap (V_1 \times V_2)$ . The parameter  $G$  may be omitted for the input graph.

We assume that the input graph has an adjacency list representation: each neighbor set has a fixed ordering,  $\Gamma(u) = \{v'_1, \dots, v'_{\deg(u)}\}$ ; this ordering may be arbitrarily (e.g., not necessarily sorted by vertex IDs). Many of the algorithms in this paper are based on partitioning the neighbor-list into balanced-size blocks. For  $\Delta \in [n]$  and  $u \in V$  such that  $\deg(u) \geq \Delta$ , let  $\Gamma_{\Delta, 1}(u), \dots, \Gamma_{\Delta, \Theta(\deg(u)/\Delta)}(u)$  be blocks of neighbors obtained by partitioning  $\Gamma(u)$  into consecutive parts. Each block is of size  $\Delta$ , except possibly for the last block that is allowed to contain up to  $2\Delta$  vertices.

**Local Computation Algorithms.** We adopt the definition of LCAs by Rubinfeld et al. [35]. A local algorithm has access to the *adjacency list oracle*  $\mathcal{O}^G$  which provides answers to the following probes (in a single step):

- **Neighbor probes:** Given a vertex  $v \in V$  and an index  $i$ , the  $i^{\text{th}}$  neighbor of  $v$  is returned if  $i \leq \deg(v)$ . Otherwise,  $\perp$  is returned. The orderings of neighbor sets are fixed in advance, but can be arbitrary.
- **Degree probes:** Given a vertex  $v \in V$ , return  $\deg(v)$ . This probe type is defined for convenience, and can alternatively be implemented via a binary search using  $O(\log n)$  NEIGHBOR probes.

<sup>11</sup> We do *not* require IDs to be a bijection  $V \rightarrow [n]$  as in other LCA papers.

- **Adjacency probes:** Given an ordered pair  $\langle u, v \rangle$ , if  $v \in \Gamma(u)$  then the index  $i$  such that  $v$  is the  $i^{\text{th}}$  neighbor of  $u$ . Otherwise,  $\perp$  is returned.

► **Definition 4** (LCA for Graph Spanners). An LCA  $\mathcal{A}$  for graph *spanners* is a (randomized) algorithm with the following properties.  $\mathcal{A}$  has access to the adjacency list oracle  $\mathcal{O}^G$  of the input graph  $G$ , a tape of random bits, and local read-write computation memory. When given an input (query) edge  $(u, v) \in E$ ,  $\mathcal{A}$  accesses  $\mathcal{O}^G$  by making probes, then returns YES if  $(u, v)$  is in the spanner  $H$ , or returns NO otherwise. This answer must only depend on the query  $(u, v)$ , the graph  $G$ , and the random bits. For a fixed tape of random bits, the answers given by  $\mathcal{A}$  to all possible edge queries, must be consistent with one particular sparse spanner.

The main complexity measures of the LCA for graph spanners are the size and stretch of the output spanner, as well as the probe complexity of the LCA, defined as the maximum number of probes that the algorithm makes on  $\mathcal{O}^G$  to return an answer for a single input edge. Informally speaking, imagine  $m$  instances of the same LCA, each of which is given an edge of  $G$  as a query, while the *shared* random tape is broadcasted to all. Each instance decides if its query edge is in the subgraph by making probes to  $\mathcal{O}^G$  and inspecting the random tape, but may not communicate with one another by any means. The LCA succeeds for the input graph  $G$  and the random tape if the collectively-constructed subgraph is a desired spanner. All the algorithms in this paper are randomized and, for any input graph, succeed with high probability  $1 - 1/n^c$  over the random tape.

**Paper Organization.** In Section 2 and 3 we describe our results for 3 and 5-spanners in general graphs. For simplicity, we first describe all our randomized algorithms as using full independence, then in Section 4, we explain how these algorithms can be implemented using a seed of poly-logarithmic number of random bits).

**Clarification.** Throughout we use the term “spanner construction” when describing how to construct our spanners. These construction algorithms are used only to define the unique spanner, based on which the LCA makes its decisions: we never construct the full, global spanner at any point.

## 2 LCA for 3-Spanners

In this section, we present the 3-spanner LCA with probe complexity of  $\tilde{O}(n^{3/4})$ . We begin in Section 2.1 by establishing some observations that allow us to “take care” of different types of edges separately based on the degrees of their endpoints. In Section 2.2-2.3 we provide constructions that take care of each type of edges; the analysis of stretch, probe complexity and spanner size for each case is included in their respective sections. We establish our final LCA for 3-spanners in Section 2.4.

### 2.1 Edge classification

► **Definition 5** (Subgraphs taking care of edges). For stretch parameter  $k$  and set of edges  $E' \subseteq E$ , we say that the subgraph  $H' \subseteq G$  *takes care* of  $E'$  if for every  $(u, v) \in E'$ ,  $\text{dist}(u, v, H') \leq k$ .

Observe that if we have a collection of subgraphs  $H_i$ 's such that every edge in  $(u, v) \in E$  is taken care by at least one  $H_i$ , then the union  $H$  of the  $H_i$ 's constitutes a  $k$ -spanner for  $G$ .

► **Observation 6** (Spanner construction by combining subgraphs). *For a collection of subsets  $E_1, \dots, E_\ell \subseteq E$  where  $\cup_{i \in [\ell]} E_i = E$ , if  $H_i$  is a subgraph of  $G$  that takes care of  $E_i$ , then  $H = \cup_{i \in [\ell]} H_i$  is a  $k$ -spanner of  $G$ . Further, if we have an LCA  $\mathcal{A}_i$  for computing each  $H_i$  (i.e., deciding whether the query edge  $(u, v) \in H_i$  and reporting YES or NO accordingly), we may construct a final LCA that runs every  $\mathcal{A}_i$  and answer YES precisely when at least one of them does so. The performance of our overall LCA (number of edges, probes, or random bits) can then be bounded by the respective sum over that of  $\mathcal{A}_i$ 's.*

Note that  $H_i$  may contain edges of  $E$  that are not in  $E_i$ , thus it is necessary that the overall LCA invokes every  $\mathcal{A}_i$  even if  $\mathcal{A}_i$  does not take care of the query edge.

**Graph partitioning.** A vertex  $v$  is low-degree if  $\deg(v) \leq \sqrt{n}$ , it is high-degree if  $\deg(v) \geq \sqrt{n}$  and it is super-high degree if  $\deg(v) \geq n^{3/4}$ . Our LCA for 3-spanner assigns each edge of  $E$  into one or more of the subsets  $E_{\text{low}}$ ,  $E_{\text{high}}$ , or  $E_{\text{super}}$  based on the degrees of its endpoints, where

$$E_{\text{low}} = \{(u, v) \in E \mid \min\{\deg(u), \deg(v)\} \leq \sqrt{n}\},$$

$$E_{\text{high}} = \{(u, v) \in E \mid \sqrt{n} < \min\{\deg(u), \deg(v)\} \leq n^{3/4}\}, \text{ and } E_{\text{super}} = E \setminus (E_{\text{low}} \cup E_{\text{high}}).$$

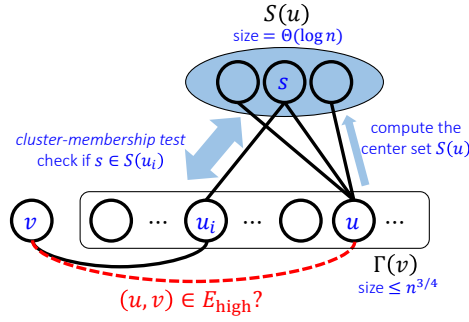
Because vertices of degree at most  $\sqrt{n}$  have  $O(n \cdot \sqrt{n}) = O(n^{3/2})$  incident edges in total, we may afford to keep all these edges, letting  $H_{\text{low}} = (V, E_{\text{low}})$ . Thus, an LCA simply needs to check the degrees of both endpoints (via DEGREE probes), and answer YES precisely when both (or in fact, even one) have degrees at most  $\sqrt{n}$ . From now on, assume that  $\deg(u), \deg(v) \geq \sqrt{n}$ .

## 2.2 3-spanner for the edges $E_{\text{high}}$

We pick a random *center* set  $S$  of size  $O(\sqrt{n} \log n)$  by sampling vertex  $v \in V$  into  $S$  independently with probability  $p = \Theta((\log n)/\sqrt{n})$ . For now, we assume that given an ID of a vertex  $v$ , we can decide in  $O(1)$  time if  $v \in S$ . At the end of the section, we describe how to implement this using a seed of  $O(\log n)$  random bits. For each endpoint  $v$  of  $E_{\text{high}}$ , let  $S(v) = \Gamma'(v) \cap S$  where  $\Gamma'(v)$  is the set of the first  $\sqrt{n}$  neighbors of  $v$  in  $\Gamma(v)$ . By Chernoff bound we have that  $|S(v)| = \Theta(\log n)$  (and in particular,  $S(v)$  is non-empty). We call  $S(v)$  the *multiple-center* set of  $v$ . The algorithm adds to  $H_{\text{high}}$  the edges  $(v, s)$  connecting  $v$  to each of its centers  $s \in S(v)$ . This adds a total of  $O(n \log n)$  edges.

Next, for every  $v$  with  $\deg(v) = O(n^{3/4})$ , the algorithm traverses its neighbor list  $\Gamma(v) = \{u_1, \dots, u_\ell\}$  and adds the edges  $(u_i, v) \in E_{\text{high}}$  to the spanner  $H_{\text{high}}$  only if  $u_i$  belongs to a *new* cluster; i.e.,  $u_i$  has a center  $s \in S(u_i)$  that no previous neighbor  $u_j$ ,  $j < i$ , has as its center in  $S(u_j)$ . Since the algorithm adds an edge whenever a new center is revealed and there are  $O(\sqrt{n} \log n)$  centers, the total number of edges added to the spanner is  $O(n^{3/2} \log n)$ .

We next describe the LCA that, given an edge  $(u, v) \in E_{\text{high}}$ , says YES iff  $(u, v) \in H_{\text{high}}$ . We assume throughout that  $\deg(v) \leq \deg(u)$ , so  $\deg(v) = O(n^{3/4})$ . First, by probing for the first  $\sqrt{n}$  neighbors of  $u$  and  $v$ , one can compute the center-sets  $S(u)$  and  $S(v)$  each containing  $O(\log n)$  centers in  $S$ . Next, the algorithm probes for all of  $v$ 's neighbors  $\Gamma(v) = \{u_1, \dots, u_j = u, \dots, u_\ell\}$ . For every neighbor  $u_i$  appearing before  $u$  in  $\Gamma(v)$ , i.e., for every  $i < j$ , and for every center  $s \in S(u)$ , the algorithm makes a *cluster-membership test* for  $s$  and  $u_i$ . This cluster-membership test can be answered by making a single ADJACENCY probe on the pair  $\langle u_i, s \rangle$ , namely  $s \in S(u_i)$  only if  $s$  is among the first  $\sqrt{n}$  neighbors of  $u_i$ . Eventually, the algorithm  $\mathcal{A}_{\text{high}}$  answers YES only if there exists  $s' \in S(u)$  such that  $s' \notin \bigcup_{i=1}^{j-1} S(u_i)$ . It is straightforward to verify that the probe complexity is  $\tilde{O}(\deg(u) + \sqrt{n}) = O(n^{3/4})$ .



■ **Figure 1** Illustration for the local construction of  $H_{\text{high}}$ .

Finally, we show that  $H_{\text{high}}$  is indeed a 3-spanner. For every edge  $(u, v)$  not added to the spanner, let  $s \in S(u)$  and let  $u_i$  be the first vertex in  $\Gamma(v)$  satisfying  $s \in S(u_i)$ . By construction,  $(u_i, v) \in H_{\text{high}}$  and also the edges  $(u_i, s)$  and  $(u, s)$  are in the spanner  $H_{\text{high}}$ , providing a path of length 3 in  $H_{\text{high}}$ .

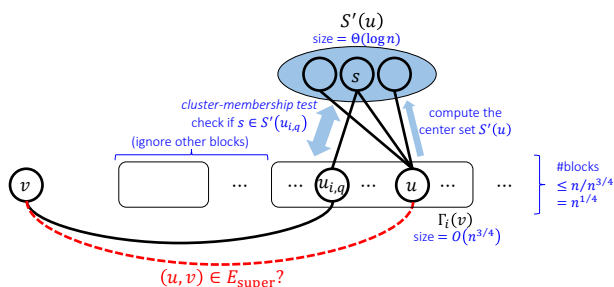
### 2.3 3-spanner for the edges $E_{\text{super}}$

We proceed by describing the construction of the 3-spanner  $H_{\text{super}}$  that takes care of the edges  $E_{\text{super}}$ . Let  $S'$  be a collection of  $O(n^{1/4} \log n)$  centers obtained by sampling each  $v \in V$  independently with probability  $p' = \Theta((\log n)/n^{3/4})$ . For each vertex  $v$ , define its center set  $S'(v)$  to be the members of  $S'$  among the first  $n^{3/4}$  neighbors of  $v$ , and if  $\deg(v) \leq n^{3/4}$ , then  $S'(v) = S' \cap \Gamma(v)$ . First, as in the construction of  $H_{\text{high}}$ , the algorithm connects each  $v$  to each of its centers by adding the edges  $(u, s)$  for every  $u$  and  $s \in S'(u)$  to the spanner  $H_{\text{super}}$ .

Consider a vertex  $v$  and divide its neighbor list into consecutive *blocks*  $\Gamma_1(v), \dots, \Gamma_\ell(v)$ , each of size  $n^{3/4}$  (expect perhaps for the last block). In every block  $\Gamma_i(v) = \{u_{i,1}, \dots, u_{i,\ell'}\}$ , the algorithm adds the edge  $(v, u_{i,j})$  to the spanner  $H_{\text{super}}$  only if  $u_{i,j}$  belongs to a *new* cluster with respect to all other vertices that appear before it in that block. Formally, the edge  $(v, u_{i,j})$  is added iff there exists  $s \in S'(u_{i,j})$  such that  $s \notin \bigcup_{q \leq j-1} S'(u_{i,q})$ . This completes the description of the construction. Observe that within each block, the LCA adds an edge for each new center. W.h.p., there are  $O(n/n^{3/4}) = O(n^{1/4})$  blocks and  $|S'| = O(n^{1/4} \log n)$  centers, so  $O(\sqrt{n} \log n)$  edges are added for each  $v$ , yielding a spanner of size  $O(n^{3/2} \log n)$ .

The LCA  $\mathcal{A}_{\text{super}}$  is very similar to  $\mathcal{A}_{\text{high}}$ : the main distinction is that given an edge  $(u, v)$  with  $\deg(u) \geq n^{3/4}$ , the algorithm  $\mathcal{A}_{\text{super}}$  will probe only for the block  $\Gamma_i(v) = \{u_{i,1}, \dots, u_{i,j} = u, u_{i,\ell'}\}$  to which  $v$  belongs, and will make its decision only based on that block. By probing for the degree of  $v$ , and the index  $j$  such that  $u$  is the  $j^{\text{th}}$  neighbor of  $v$ , one can compute the block  $\Gamma_i(v)$  by making  $n^{3/4}$  NEIGHBOR probes. In addition, by probing for the first  $n^{3/4}$  neighbors of both  $u$  and  $v$ , one can compute the multiple-center sets  $S'(u)$  and  $S'(v)$ . Finally, the algorithm applies a cluster-membership test for each pair  $s \in S'(u)$  and  $u_{i,q}$  for  $q \leq j-1$ . It returns YES only if there exists  $s \notin \bigcup_{q \leq j-1} S'(u_{i,q})$ . Hence, the number of probes made by the LCA is w.h.p. bounded by  $|\Gamma_i(v)| \cdot |S'(u)| = O(n^{3/4} \log n)$ .

We now show that  $H_{\text{super}}$  is a 3-spanner for the edges  $H_{\text{super}}$ . Let  $(u, v)$  be such that  $\deg(u) \geq n^{3/4}$  and let  $\Gamma_i(v)$  be the block in  $\Gamma(v)$  to which  $u$  belongs. Since  $\deg(u) \geq n^{3/4}$ , w.h.p.  $|S'(u)| = \Theta(\log n)$ . Assume that  $(u, v) \notin H_{\text{super}}$ . Fix  $s \in S'(u)$  and let  $u_{i,q}$  be the first vertex in  $\Gamma_i(v)$  that belongs to the cluster of  $s$ . Since  $(u, v) \notin H_{\text{super}}$ , such a vertex  $u_{i,q}$  is guaranteed to exist. The spanner  $H_{\text{super}}$  contains the edges  $(s, u)$ ,  $(s, u_{i,q})$  and  $(v, u_{i,q})$ , thus containing a path of length 3 between  $u$  and  $v$ .



■ **Figure 2** Illustration for the local construction of  $H_{\text{super}}$ .

## 2.4 The Final LCA

Given an edge  $(u, v)$  the algorithm says YES if one of the following holds:

- $\deg(u), \deg(v) \leq \sqrt{n}$ .
- $u \in S(v) \cup S'(v)$  (or vice versa).
- the local algorithm  $\mathcal{A}_{\text{high}}$  says YES on edge  $(u, v)$ .
- the local algorithm  $\mathcal{A}_{\text{super}}$  says YES on edge  $(u, v)$ .

This completes the 3-spanner LCA from Theorem 1.

**Missing piece: computing centers in the LCA model.** In the LCA model, we do not generate the entire set  $S$  (or  $S'$ ) up front. Instead, we may verify whether  $v \in S$  on-the-fly using  $v$ 's ID by, e.g., applying a random map (chosen according to the given random tape) from  $v$ 's ID to  $\{0, 1\}$  with expectation  $p$ . In fact, this hitting set argument does not require full independence – the discussion on reducing the amount of random bits is given in Section 4, but for now we formalize it as the following observation.

► **Observation 7** (Local Computation of Centers). *Let  $S$  be a center set obtained by placing each vertex into  $S$  independently with probability  $p = \Theta(\log n / \Delta)$ . W.h.p.,  $S$  forms a hitting set for the collection of neighbor sets of all vertices of degree at least  $\Delta$ . Further, under the LCA model, we may check whether  $v \in S$  locally without making any probes.*

## 3 LCA for 5-Spanners

We now consider LCAs for 5-spanners, aiming for spanners of size  $\tilde{O}(n^{4/3})$  with probe complexity  $\tilde{O}(n^{5/6})$ . We start by noting that the construction of  $H_{\text{super}}$  for the 3-spanners in fact gives for every  $r \geq 1$ , a 3-spanner of size  $\tilde{O}(n^{1+1/r})$  for the subset of edges  $(u, v)$  with  $\min\{\deg(u), \deg(v)\} \geq n^{1-1/(2r)}$ : this is achieved by instead setting the threshold for super-high degree at  $n^{1-1/(2r)}$ , pick  $|S'| = \tilde{O}(n^{1/(2r)})$  centers, and use block size  $n^{1-1/(2r)}$ . The probe complexity for querying the spanner is  $\tilde{O}(n^{1-1/(2r)})$ . For 5-spanner, by taking  $r = 3$ , one takes care of all edges  $(u, v)$  with  $\max\{\deg(u), \deg(v)\} \geq n^{5/6}$ .

Let  $\Delta_{\text{low}} = n^{1/r}$ ,  $\Delta_{\text{med}} = n^{1/2-1/(2r)}$  and  $\Delta_{\text{super}} = n^{1-1/(2r)}$ . For the purpose of constructing 5-spanners for general graphs, we let  $r = 3$ , simplifying the thresholds to  $\Delta_{\text{low}} = \Delta_{\text{med}} = n^{1/3}$  and  $\Delta_{\text{super}} = n^{5/6}$ .) Again, we may afford to keep all edges incident to some vertex of degree at most  $\Delta_{\text{low}}$ .

For integers  $a \leq b$ , let  $V_{[a,b]} = \{v \in V(G) \mid \deg(v) \in [a, b]\}$ . We will design a subgraph  $H \subseteq G$  that will take care of the remaining edges  $E_{\text{med}} = E(V_{[\Delta_{\text{med}}, \Delta_{\text{super}]}, V_{[\Delta_{\text{med}}, \Delta_{\text{super}]})}$ .

■ **Table 2** Edge categorization for the construction of 5-spanners.

Subset	Criteria	# Edges	Probe Complexity
$E_{\text{low}}$	$(u, v) \in E(V, V_{[1, \Delta_{\text{low}}]})$	$O(n \cdot \Delta_{\text{low}}) = O(n^{1+\frac{1}{r}})$	$O(1)$
$E_{\text{bckt}}$	$(u, v) \in E(V_{\text{dsrt}}, V_{\text{dsrt}})$	$O(\frac{n^2 \log^2 n}{\Delta_{\text{med}}^2}) = O(n^{1+\frac{1}{r}} \log^2 n)$	$O((\Delta_{\text{super}} + \Delta_{\text{med}}^2) \log^2 n) = O(n^{1-\frac{1}{2r}} \log^2 n)$
$E_{\text{rep}}$	$(u, v) \in E(V_{[\Delta_{\text{med}}, \Delta_{\text{super}}]}, V_{\text{crwd}})$	$O(\frac{n^2}{\Delta_{\text{super}}} \cdot \log n) = O(n^{1+\frac{1}{r}} \log n)$	$O(\Delta_{\text{super}} \log^3 n) = O(n^{1-\frac{1}{2r}} \log^3 n)$
$E_{\text{super}}$	$(u, v) \in E(V, V_{[\Delta_{\text{super}}, n]})$	$O(\frac{n^2 \log n}{\Delta_{\text{super}}}) = O(n^{1+\frac{1}{r}} \log n)$	$O(\Delta_{\text{super}} \log n) = O(n^{1-\frac{1}{2r}} \log n)$

► **Definition 8** (Deserted and Crowded vertices). A vertex  $v \in V_{[\Delta_{\text{med}}, \Delta_{\text{super}}]}$  is *deserted* if at least half of its neighbors in  $\Gamma_{\Delta_{\text{med}}, 1}(v)$  are of degree at most  $\Delta_{\text{super}}$ ; i.e.,  $|\Gamma_{\Delta_{\text{med}}, 1}(v) \cap V_{[1, \Delta_{\text{super}}]}| \geq \Delta_{\text{med}}/2$ . Otherwise, the vertex is *crowded*.

**Criteria for edges.** We aim to take care of edges for which both endpoints are in  $V_{[\Delta_{\text{med}}, \Delta_{\text{super}}]}$ . To categorize our edges for the purpose of constructing 5-spanners, we need the following partition of these vertices.

Let  $V_{\text{dsrt}}$  (resp.,  $V_{\text{crwd}}$ ) be the set of deserted (resp., crowded) vertices in  $V_{[\Delta_{\text{med}}, \Delta_{\text{super}}]}$ . Given a vertex, we can verify whether it is in any of these sets using  $O(\Delta_{\text{med}})$  probes by checking the degrees of  $v$  and each vertex in  $\Gamma_{\Delta_{\text{med}}, 1}(v)$ . We then assign each  $(u, v) \in E$  into one of the four cases {low, bckt, rep, super} as given in Table 2. It is straightforward to verify that when  $\Delta_{\text{low}} = \Delta_{\text{med}}$  (namely when we choose  $r = 3$ , which also yields the required performance), these four cases take care of all edges in  $E$ . We note that  $H_{\text{rep}}$  assumes that  $H_{\text{super}}$  is included:  $E_{\text{rep}}$  is taken care by  $H_{\text{rep}} \cup H_{\text{super}}$ , not by  $H_{\text{rep}}$  alone.

**LCA for  $E_{\text{bckt}}$ : the cluster partitioning method.** The algorithm is as follows.

- Only vertices of degree at most  $\Delta_{\text{super}}$  are chosen to be in  $S$  with probability  $p = \Theta((\log n)/\Delta_{\text{med}})$ . Since at least half the vertices in  $\Gamma_{\Delta_{\text{med}}, 1}(v)$  for any  $v \in V_{\text{dsrt}}$  have degree smaller than  $\Delta_{\text{super}}$ , we have that w.h.p.  $|S(v)| = \Theta(\log n)$  the cluster-membership test can be done with constant number of probes. Let us denote by  $C(s) = \{s\} \cup \{v : s \in S(v)\}$  the cluster of center  $s$ .
- The partitioning of clusters into buckets is defined in a consistent way (regardless of the given query edge); for instance, create a list of vertices in the cluster, sort them according to their IDs, divide the list into buckets of size  $\Delta_{\text{med}}$  possibly except for the last one. Note that we partition  $C(s)$  and  $C(t)$  separately – we do not combine their elements. Similarly, once we obtain buckets containing  $u$  and  $v$ , the order in which we check the adjacency of  $u'$  and  $v'$  must be consistent. To this end, define the ID of an edge  $(u, v)$  as  $(\text{ID}(u), \text{ID}(v))$ , where the comparison between edge IDs is lexicographic. Thus, this step only adds the edge of minimum ID between the two clusters.
- We also set the precondition  $(u, v) \in E(V_{[\Delta_{\text{med}}, n]}, V_{[\Delta_{\text{med}}, n]})$ , and consistently only allow candidate pairs  $(u', v') \in E(V_{[\Delta_{\text{med}}, n]}, V_{[\Delta_{\text{med}}, n]})$ , to ensure that the lexicographically first edge of this exact specification is added if one exists. We do not restrict to  $E_{\text{bckt}}$ , which require both endpoints to be deserted vertices, because checking whether  $(u', v') \in E_{\text{bckt}}$  would take  $\Theta(\Delta_{\text{med}})$  probes instead of constant probes. We restrict to edges whose endpoints have degrees at least  $\Delta_{\text{med}}$  instead of considering the entire  $E$  so that  $S$  would be well-defined.



**Local construction of  $H_{\text{bckt}}$ .** Each  $v \in V_{[1, \Delta_{\text{super}}]}$  is added to  $S$  with probability  $p = \Theta(\log n / \Delta_{\text{med}})$ .

(A) If  $u \in S(v)$  or  $v \in S(u)$ , answer YES.

(B) If  $(u, v) \in E(V_{[\Delta_{\text{med}}, n]}, V_{[\Delta_{\text{med}}, n]})$ :

- Compute  $S(u)$  and  $S(v)$  by iterating through  $\Gamma_{\Delta_{\text{med}}, 1}(u)$  and  $\Gamma_{\Delta_{\text{med}}, 1}(v)$ .
- For each pair of  $s \in S(u)$  and  $t \in S(v)$ :
  - Partition each of the clusters  $C(s)$  and  $C(t)$  into buckets of size (mostly)  $\Delta_{\text{med}}$ . Denote the buckets containing  $u$  and  $v$  by  $\text{Bucket}(u, s)$  and  $\text{Bucket}(v, t)$ , respectively.
  - Iterate through each pair of  $u' \in \text{Bucket}(u, s)$  and  $v' \in \text{Bucket}(v, t)$  and check if  $(u', v') \in E(V_{[\Delta_{\text{med}}, n]}, V_{[\Delta_{\text{med}}, n]})$ . Answer YES if the edge of minimum ID found is  $(u', v') = (u, v)$ .

► **Lemma 9.** For  $1 \leq \Delta_{\text{med}} \leq \sqrt{n} \leq \Delta_{\text{super}} \leq n$ , there exists a subgraph  $H_{\text{bckt}} \subseteq G$  such that w.h.p.:

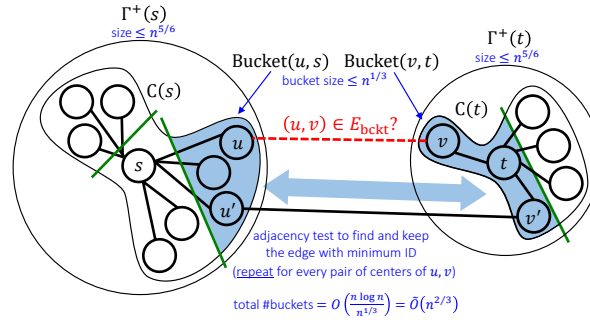
- (i)  $H_{\text{bckt}}$  has  $O(\frac{n^2 \log^2 n}{\Delta_{\text{med}}^2})$  edges,
- (ii)  $H_{\text{bckt}}$  takes care of  $E_{\text{bckt}}$ ; that is, for every  $(u, v) \in H_{\text{bckt}}$ ,  $\text{dist}(u, v, H_{\text{bckt}}) \leq 5$ , and
- (iii) for a given edge  $(u, v) \in E$ , one can test if  $(u, v) \in H_{\text{bckt}}$  by making  $O((\Delta_{\text{super}} + \Delta_{\text{med}}^2) \log^2 n)$  probes.

**Proof.**

- (i) **Size.** In (A) we add  $|S(v)| = \Theta(\log n)$  edges for each  $v \in V_{\text{dsrt}}$ , which constitutes to  $O(n \log n)$  edges in total. In (B), we add one edge between each pair of buckets. We now compute the total number of buckets. The total size of clusters  $\sum_{s \in S} |C(s)| \leq |S| + \sum_{v \in V_{[\Delta_{\text{med}}, n]}} |S(v)| = O(n \log n)$ , so there can be up to  $O((n \log n) / \Delta_{\text{med}})$  full buckets of size  $\Delta_{\text{med}}$ . As buckets are formed by partitioning  $|S|$  clusters, there are up to  $|S| = \Theta((n \log n) / \Delta_{\text{med}})$  remainder buckets of size less than  $\Delta_{\text{med}}$ . Thus, there are  $\Theta((n \log n) / \Delta_{\text{med}})$  buckets, and  $O(((n \log n) / \Delta_{\text{med}})^2)$  edges are added in (B).
- (ii) **Stretch.** Suppose that  $(u, v)$  is omitted. Fix centers  $s \in S(u)$  and  $t \in S(v)$ , then the lexicographically-first edge  $(u', v') \in E(\text{Bucket}(u, s), \text{Bucket}(v, t))$  must have been added to  $H_{\text{bckt}}$ , forming the path  $\langle u, s, u', v', t, v \rangle$  (or shorter, if there are repeated vertices), yielding  $\text{dist}(u, v, H_{\text{bckt}}) \leq 5$ .
- (iii) **Probes.** Computing  $S(u)$  and  $S(v)$  takes  $O(\Delta_{\text{med}})$  probes. For each pairs of centers, we scan through the entire neighbor-lists  $\Gamma(s)$  and  $\Gamma(t)$  and collect all vertices in their respective clusters. This takes  $O(\Delta_{\text{super}})$  probes each because we restrict to centers of degree at most  $\Delta_{\text{super}}$ . Given the clusters, we identify the buckets containing  $u$  and  $v$  each of size  $O(\Delta_{\text{med}})$ . We then check through candidates  $(u', v')$  between these buckets, taking  $O(\Delta_{\text{med}}^2)$  ADJACENCY probes. So, each pair of centers requires  $O(\Delta_{\text{super}} + \Delta_{\text{med}}^2)$  total probes. We repeat the process for  $|S(u)| \cdot |S(v)| = O(\log^2 n)$  pairs of centers w.h.p., yielding the claimed probe complexity. ◀

**LCA for  $E_{\text{rep}}$ : the Representative method.** We first explain the computation of the representative set  $\text{Reps}(v)$  for a crowded vertex  $v \in V_{\text{crwd}}$ , i.e., a collection of neighbors of  $v$  that have degree at least  $n^{5/6}$ . Using the random bits and the vertex ID, we sample a set  $R_v$  of  $\Theta(\log n)$  (not necessarily distinct) indices in  $[\Delta_{\text{med}}]$  at random (for details, see Sec. 4). Denote the neighbor-list of  $v$  by  $\{x'_1, \dots, x'_{\deg(v)}\}$ , then define  $\text{Reps}(v) = \{x'_i : i \in R_v \text{ and } \deg(x'_i) \geq \Delta_{\text{super}}\}$ . Then since at least half of the vertices in  $\Gamma_{\Delta_{\text{med}}, 1}(v)$  are of degree





■ **Figure 3** Illustration for the local construction of  $H_{\text{bckt}}$ . Green lines show the partition of clusters into buckets.

at least  $\Delta_{\text{super}}$ , w.h.p.  $\text{Reps}(v) \neq \emptyset$ . For consistency, we allow the same definition for  $\text{Reps}(v)$  for any  $v \in V_{[\Delta_{\text{med}}, n]}$  as well, even if it may result in empty sets of representatives. Hence computing  $\text{Reps}(v)$  takes  $O(\log n)$  probes<sup>12</sup>.

Let  $E_{\text{super}} = \{(u, v) \in E \mid \max\{\deg(u), \deg(v)\} \geq n^{5/6}\}$  and apply the 3-spanner algorithm of Sec. 2 to construct a subgraph  $H_{\text{super}}$  that takes care of the edges  $E_{\text{super}}$ . To construct  $H_{\text{super}}$  the algorithm (fully described<sup>13</sup> in Sec. 2) samples a set  $S'$  of centers by picking each  $v \in V$  independently with probability  $O(\log n/n^{5/6})$ . For every  $v$  with  $\deg(v) \geq n^{5/6}$ , let  $S'(v)$  be the sampled neighbors in  $S' \cap \Gamma_1(v)$  where  $\Gamma_1(v)$  is the first block of size  $n^{5/6}$  in  $\Gamma(v)$ . This allows us to check membership to a cluster of  $s \in S'$  using a single adjacency probe. The idea would be to extend the 1-radius clusters of  $S'$  by one additional layer consisting of the crowded vertices connected to the cluster via their representatives.

For convenience, for a crowded  $v$ , define  $RS(v) = \cup_{x' \in \text{Reps}(v)} S'(x')$ , the set of (multiple) centers of any of  $v$ 's representatives. Observe that by adding the edge  $(v, x')$  to  $H_{\text{rep}}$  for every  $x' \in \text{Reps}(v)$ , it yields that  $\text{dist}(v, s, H_{\text{rep}} \cup H_{\text{super}}) \leq 2$  for any  $s \in RS(v)$ .

Consider the query  $(u, v)$ , and suppose that  $v = v'_i$  is the  $i^{\text{th}}$  neighbor in  $u$ 's neighbor-list,  $\Gamma(u) = \{v'_1, \dots, v'_{\deg(u)}\}$ . We then add  $(u, v)$  to  $H_{\text{rep}}$  if and only if  $v$  introduces a new center through some representative; that is,  $RS(v'_i) \setminus \cup_{j < i} RS(v'_j) \neq \emptyset$ . To verify this condition locally, we first compute  $RS(v)$ , and for each of  $\{v'_j\}_{j < i}$ ,  $\text{Reps}(v'_j)$ . Then, we discard  $(u, v)$  if for every center  $s \in RS(v)$ , there exists  $x$  and  $v'_j$  where  $x \in \text{Reps}(v'_j)$  and  $s \in S'(x)$ ; the last condition takes constant probes to verify. This gives the full LCA for constructing  $H_{\text{rep}}$  below.

**Local construction of  $H_{\text{rep}}$ .** Each  $v \in V$  is added to  $S'$  with probability  $p = \Theta((\log n)/\Delta_{\text{super}})$ .

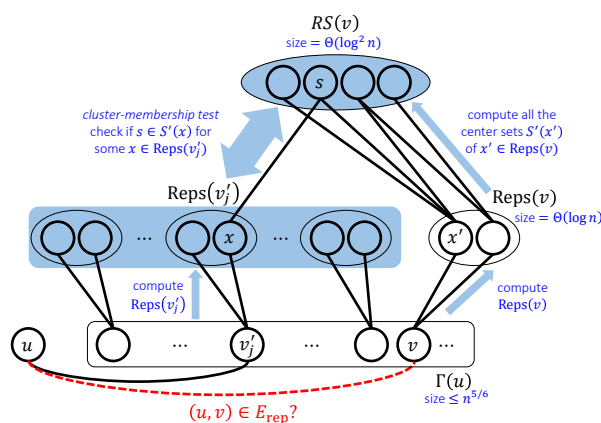
(A) If  $v \in V_{[\Delta_{\text{med}}, \Delta_{\text{super}}]}$  and  $u \in \text{Reps}(v)$ , answer YES.

(B) If  $u, v \in V_{[\Delta_{\text{med}}, \Delta_{\text{super}}]}$ :

- Compute  $RS(v)$ .
- Denote the neighbor-list of  $u$  by  $\{v'_1, \dots, v'_{\deg(u)}\}$ ; identify  $i$  such that  $v = v'_i$ .
- For each vertex  $w \in \{v'_1, \dots, v'_{i-1}\}$ , if  $w \in V_{[\Delta_{\text{med}}, \Delta_{\text{super}}]}$ , compute  $\text{Reps}(w)$ .
- For each  $s \in RS(v)$ , iterate to check for a vertex  $x$  in any of the  $\text{Reps}(w)$ 's obtained above, such that  $s \in S'(x)$ . Answer YES if there exists a vertex  $s$  where no such  $x$  exists.

<sup>12</sup> The naïve solution traverses the entire  $\Delta_{\text{med}}$  first neighbors of  $v$  which is too costly.

<sup>13</sup> Upon replacing the degree threshold of  $n^{3/4}$  with  $n^{5/6}$ .



■ **Figure 4** Illustration for the local construction of  $H_{\text{rep}}$ .

► **Lemma 10.** For  $1 \leq \Delta_{\text{med}} \leq \Delta_{\text{super}} \leq n$ , there exists a subgraph  $H_{\text{rep}} \subseteq G$  such that w.h.p.:

- (i)  $H_{\text{rep}}$  has  $O(n^2/\Delta_{\text{super}} \cdot \log n)$  edges,
- (ii)  $H_{\text{rep}} \cup H_{\text{super}}$  takes care of  $E_{\text{rep}}$ ; that is, for every  $(u, v) \in E_{\text{rep}}$ ,  $\text{dist}(u, v, H_{\text{rep}} \cup H_{\text{super}}) \leq 3$ , and
- (iii) for a given edge  $(u, v) \in E$ , one can test if  $(u, v) \in H_{\text{rep}}$  by making  $O(\Delta_{\text{super}} \log^3 n)$  probes.

**Proof.**

- (i) **Size.** W.h.p., in (A) we add at most  $\sum_{v \in V_{[\Delta_{\text{med}}, \Delta_{\text{super}}]}} |\text{Reps}(v)| \leq n \cdot O(\log n) = O(n \log n)$ . Similarly to the analysis of  $H_{\text{high}}$ , in (B) we add  $|S'| = O((n \log n)/\Delta_{\text{super}})$  edges per vertex  $u$ , so  $|E(H_{\text{rep}})| = O(n^2/\Delta_{\text{super}} \cdot \log n)$ .
- (ii) **Stretch.** This claim follows from the argument given in the overview, and is similar to the analysis of  $H_{\text{high}}$ .
- (iii) **Probes.** Computing  $RS(v)$  takes  $O(\log n) \cdot \Delta_{\text{super}} = O(\Delta_{\text{super}} \log n)$  (recall that we only check  $\Gamma_{\Delta_{\text{super}}, 1}$  of each representative). Note also that  $|RS(v)| = O(\log^2 n)$  since  $v$  has  $O(\log n)$  representative, each of which belongs to  $\Theta(\log n)$  clusters. Computing  $\text{Reps}$  for each neighbor  $w \in \{v'_j\}_{j < i}$  of  $u$  takes  $O(\log n)$  probes each, which is  $O(\Delta_{\text{super}} \log n)$  in total since  $\deg(u) \leq \Delta_{\text{super}}$ . This also introduces up to  $\Delta_{\text{super}} \cdot O(\log n)$  representatives in total. Checking whether each of the  $O(\log^2 n)$  centers in  $RS(v)$  is a center of each of these  $O(\Delta_{\text{super}} \log n)$  representative takes, in total w.h.p.,  $O(\Delta_{\text{super}} \log^3 n)$  probes. ◀

**Final 5-spanner results.** To obtain an LCA for 5-spanners, we again invoke all of our LCAs for the four cases. Applying Lemma 9 and 10, we obtain the following LCA result for 5-spanner in general graphs.

► **Theorem 11.** For every  $n$ -vertex simple undirected graph  $G = (V, E)$  there exists an LCA for 5-spanner with  $O(n^{4/3} \log^2 n)$  edges and probe complexity  $O(n^{5/6} \log^3 n)$ .

Again, by combining results for larger degrees, we obtain an LCA for 5-spanners with smaller sizes on graphs with minimum degree at least  $n^{1/2-1/(2r)}$ .

► **Theorem 12.** For every  $r \geq 1$  and  $n$ -vertex simple undirected graph  $G = (V, E)$  with minimum degree at least  $n^{1/2-1/(2r)}$ , there exists a (randomized) LCA for 5-spanner with  $O(n^{1+1/r} \log^2 n)$  edges and probe complexity of  $O(n^{1-1/(2r)} \log^3 n)$ .

## 4 Bounded Independence

In this section, we show that all our LCA constructions succeed w.h.p. using  $\Theta(\log n)$ -wise independent hash functions which only require  $\Theta(\log^2 n)$  random bits. We use the following standard notion of  $d$ -wise independent hash functions as in [38]. In particular, our algorithms use the explicit construction of  $\mathcal{H}$  by [38], with the parameters as stated in Lemma 14.

► **Definition 13.** For  $N, M, d \in \mathbb{N}$  such that  $d \leq N$ , a family of functions  $\mathcal{H} = \{h : [N] \rightarrow [M]\}$  is  $d$ -wise independent if for all distinct  $x_1, \dots, x_d \in [N]$ , the random variables  $h(x_1), \dots, h(x_d)$  are independent and uniformly distributed in  $[M]$  when  $h$  is chosen randomly from  $\mathcal{H}$ .

► **Lemma 14** (Corollary 3.34 in [38]). *For every  $\gamma, \beta, d \in \mathbb{N}$ , there is a family of  $d$ -wise independent functions  $\mathcal{H}_{\gamma, \beta} = \left\{ h : \{0, 1\}^\gamma \rightarrow \{0, 1\}^\beta \right\}$  such that choosing a random function from  $\mathcal{H}_{\gamma, \beta}$  takes  $d \cdot \max\{\gamma, \beta\}$  random bits, and evaluating a function from  $\mathcal{H}_{\gamma, \beta}$  takes time  $\text{poly}(\gamma, \beta, d)$ .*

Then, we exploit the following result to show the concentration of  $d$ -wise independent random variables:

► **Fact 15** (Theorem 5(III) in [36]). *If  $X$  is a sum of  $d$ -wise independent random variables, each of which is in the interval  $[0, 1]$  with  $\mu = \mathbb{E}(X)$ , then:*

- (I) *For  $\delta \leq 1$  and  $d \leq \lfloor \delta^2 \mu e^{-1/3} \rfloor$ , it holds that  $\Pr[|X - \mu| \geq \delta \mu] \leq e^{-\lfloor d/2 \rfloor}$ .*
- (II) *For  $\delta \geq 1$  and  $d = \lceil \delta \mu \rceil$ , it holds that:  $\Pr[|X - \mu| \geq \delta \mu] \leq e^{-\delta \mu/3}$ .*

**Bounded independence for hitting set procedures.** Most of our algorithms are based on the following hitting set procedure. For a given threshold  $\Delta \in [1, n]$ , each vertex flips a coin with probability  $p = (c \log n)/\Delta$  of being head and the set of all vertices with head outcome join the set of centers  $S$ . Assuming the outcome of coin flips are fully independent, by the Chernoff bound, the followings hold w.h.p.:

(HI) There are  $\Theta(pn)$  sampled vertices  $S$ .

(HII) For each vertex of degree at least  $\Delta$ , it has  $\Theta(\log n)$  centers among its first  $\Delta$  neighbors. Here we show that to satisfy properties (HI) and (HII), it is sufficient to assume that the outcomes of the coin flips are  $d$ -wise independent. By Lemma 14, to simulate  $d$ -wise independent coin flips for all vertices, the algorithm only requires  $t = \Theta(d(\log n + \log 1/p))$  random bits: more precisely, setting  $\gamma = \Theta(\log n)$  and  $\beta = \log 1/p$  (for simplicity, lets assume that  $\log 1/p$  is an integer), there exists a family of  $d$ -wise independent functions  $\mathcal{H} = \left\{ h : \{0, 1\}^{\Theta(\log n)} \rightarrow \{0, 1\}^{\log(1/p)} \right\}$  such that a random function  $h \in \mathcal{H}$  can be specified by a string of random bits of length  $t$ . In other words, each function  $h \in \mathcal{H}$  maps the ID of each vertex to the outcome of its coin flip according to a coin with bias  $p$ . Then, from a string  $\mathcal{R}$  of  $t$  random bits, the algorithm picks a function  $h_{\mathcal{R}} \in \mathcal{H}$  at random to simulate the coin flips of the vertices accordingly: the outcome of the coin flip of  $v$  is head if  $h_{\mathcal{R}}(\text{ID}(v)) = 0$  (which happens with probability  $p$ ) and the coin flips are  $d$ -wise independent. Setting  $d = c \log n$  for some constant  $c > 1$ , we prove the following:

► **Claim 16.** *If the coin flips are  $d$ -wise independent then properties (HI) and (HII) holds. Furthermore, the sequence of  $n$   $d$ -wise independent coin flips can be simulated using a string of  $O(\log^2 n)$  random bits.*

**Construction of representatives in Section 3.** The analysis above also extends to the process of computing Reprs. Each crowded vertex chooses values  $c \log n$  random indices (of its neighbor-list) in  $[\Delta_{\text{med}}]$ , each of which has probability  $1/2$  of hitting a neighbor of degree at least  $\Delta_{\text{super}}$ . Let  $\{Z_i\}_{i \in [c \log n]}$  be indicators for these events and  $Z$  denote their sum, then the expected sum  $\mathbb{E}(Z) \geq (c/2) \log n$ . Imposing  $d$ -wise independence, Fact 15(I) implies that w.h.p.,  $Z > 0$ , so the representative set is non-empty. We apply the union bound to show that  $\text{Reps}(v) \neq \emptyset$  for every  $v \in V_{\text{crwd}}$ , as desired.

---

## References

- 1 Noga Alon, Ronitt Rubinfeld, Shai Vardi, and Ning Xie. Space-efficient local computation algorithms. In *Proc. 23rd ACM-SIAM Sympos. Discrete Algs. (SODA)*, pages 1132–1139, 2012.
- 2 Baruch Awerbuch and David Peleg. Network synchronization with polylogarithmic overhead. In *Proc. 31st Annu. IEEE Sympos. Found. Comput. Sci. (FOCS)*, pages 514–522, 1990.
- 3 Baruch Awerbuch and David Peleg. Routing with polynomial communication-space trade-off. *SIAM J. Discrete Math.*, 5(2):151–162, 1992.
- 4 Surender Baswana, Sumeet Khurana, and Soumojit Sarkar. Fully dynamic randomized algorithms for graph spanners. *ACM Transactions on Algorithms (TALG)*, 8(4):35, 2012.
- 5 Surender Baswana and Sandeep Sen. A Simple and Linear Time Randomized Algorithm for Computing Sparse Spanners in Weighted Graphs. *Random Structures and Algorithms*, 30(4):532–563, 2007.
- 6 Greg Bodwin and Sebastian Krinninger. Fully Dynamic Spanners with Worst-Case Update Time. In *Proc. 24th Annu. European Sympos. Algorithms (ESA)*, pages 17:1–17:18, 2016.
- 7 Keren Censor-Hillel, Merav Parter, and Gregory Schwartzman. Derandomizing Local Distributed Algorithms under Bandwidth Restrictions. In *31st International Symposium on Distributed Computing, DISC 2017, October 16-20, 2017, Vienna, Austria*, pages 11:1–11:16, 2017.
- 8 Bilel Derbel and Cyril Gavoille. Fast deterministic distributed algorithms for sparse spanners. *Theoretical Computer Science*, 2008.
- 9 Bilel Derbel, Cyril Gavoille, and David Peleg. Deterministic distributed construction of linear stretch spanners in polylogarithmic time. In *Proc. 21st Int. Symp. Dist. Comp. (DISC)*, pages 179–192, 2007.
- 10 Bilel Derbel, Cyril Gavoille, David Peleg, and Laurent Viennot. On the locality of distributed sparse spanner construction. In *Proceedings of the Twenty-Seventh Annual ACM Symposium on Principles of Distributed Computing, PODC 2008, Toronto, Canada, August 18-21, 2008*, pages 273–282, 2008.
- 11 Bilel Derbel, Cyril Gavoille, David Peleg, and Laurent Viennot. Local computation of nearly additive spanners. In *Proc. 23rd Int. Symp. Dist. Comp. (DISC)*, 2009.
- 12 Michael Elkin. Streaming and fully dynamic centralized algorithms for constructing and maintaining sparse spanners. *ACM Transactions on Algorithms (TALG)*, 7(2):20, 2011.
- 13 Michael Elkin and Ofer Neiman. Efficient Algorithms for Constructing Very Sparse Spanners and Emulators. In *Proc. 28th ACM-SIAM Sympos. Discrete Algs. (SODA)*, pages 652–669, 2017.
- 14 P Erdős. On some extremal problems in graph theory. *Israel Journal of Mathematics*, 3(2):113–116, 1965.
- 15 Guy Even, Moti Medina, and Dana Ron. Deterministic stateless centralized local algorithms for bounded degree graphs. In *Proc. 22nd Annu. European Sympos. Algorithms (ESA)*, pages 394–405, 2014.

- 16 Mohsen Ghaffari and Jara Uitto. Sparsifying distributed algorithms with ramifications in massively parallel computation and centralized local computation. *Proc. 30th ACM-SIAM Sympos. Discrete Algs. (SODA)*, 2019.
- 17 Oded Goldreich. A brief introduction to property testing. In *Studies in Complexity and Cryptography. Miscellanea on the Interplay between Randomness and Computation*, pages 465–469. Springer, 2011.
- 18 Oded Goldreich, Shafi Goldwasser, and Dana Ron. Property testing and its connection to learning and approximation. *Journal of the ACM (JACM)*, 45(4):653–750, 1998.
- 19 Michael Kapralov and Rina Panigrahy. Spectral sparsification via random spanners. In *Proceedings of the 3rd Innovations in Theoretical Computer Science Conference*, pages 393–398, 2012.
- 20 Tali Kaufman, Michael Krivelevich, and Dana Ron. Tight bounds for testing bipartiteness in general graphs. *SIAM Journal on computing*, 33(6):1441–1483, 2004.
- 21 Christoph Lenzen and Reut Levi. A Centralized Local Algorithm for the Sparse Spanning Graph Problem. In *Proc. 45th Int. Colloq. Automata Lang. Prog. (ICALP)*, pages 87:1–87:14, 2018.
- 22 Reut Levi, Guy Moshkovitz, Dana Ron, Ronitt Rubinfeld, and Asaf Shapira. Constructing near spanning trees with few local inspections. *Random Structures & Algorithms*, 50(2):183–200, 2017.
- 23 Reut Levi and Dana Ron. A quasi-polynomial time partition oracle for graphs with an excluded minor. *ACM Transactions on Algorithms (TALG)*, 11(3):24, 2015.
- 24 Reut Levi, Dana Ron, and Ronitt Rubinfeld. Local Algorithms for Sparse Spanning Graphs. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, APPROX/RANDOM*, pages 826–842, 2014.
- 25 Reut Levi, Dana Ron, and Ronitt Rubinfeld. A local algorithm for constructing spanners in minor-free graphs. *arXiv preprint*, 2016. [arXiv:1604.07038](https://arxiv.org/abs/1604.07038).
- 26 Reut Levi, Ronitt Rubinfeld, and Anak Yodpinyanee. Local Computation Algorithms for Graphs of Non-constant Degrees. *Algorithmica*, 77(4):971–994, 2017.
- 27 Yishay Mansour, Boaz Patt-Shamir, and Shai Vardi. Constant-time local computation algorithms. In *International Workshop on Approximation and Online Algorithms*, pages 110–121, 2015.
- 28 Yishay Mansour, Aviad Rubinfeld, Shai Vardi, and Ning Xie. Converting online algorithms to local computation algorithms. In *Automata, Languages, and Programming - 39th International Colloquium, ICALP 2012, Warwick, UK, July 9-13, 2012, Proceedings, Part I*, pages 653–664. Springer, 2012.
- 29 Yishay Mansour and Shai Vardi. A local computation approximation scheme to maximum matching. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*, pages 260–273. Springer, 2013.
- 30 David Peleg. *Distributed Computing: A Locality-sensitive Approach*. SIAM, 2000.
- 31 David Peleg and Alejandro A Schäffer. Graph spanners. *Journal of graph theory*, 13(1):99–116, 1989.
- 32 David Peleg and Jeffrey D Ullman. An optimal synchronizer for the hypercube. *SIAM Journal on computing*, 18(4):740–747, 1989.
- 33 Seth Pettie. Distributed algorithms for ultrasparse spanners and linear size skeletons. *Distributed Computing*, 22(3):147–166, 2010.
- 34 Omer Reingold and Shai Vardi. New techniques and tighter bounds for local computation algorithms. *Journal of Computer and System Sciences*, 82(7):1180–1200, 2016.
- 35 Ronitt Rubinfeld, Gil Tamir, Shai Vardi, and Ning Xie. Fast Local Computation Algorithms. In *Innovations in Computer Science - ICS 2010*, pages 223–238, 2011.

- 36 Jeanette P Schmidt, Alan Siegel, and Aravind Srinivasan. Chernoff-Hoeffding bounds for applications with limited independence. *SIAM J. Discrete Math.*, 8(2):223–250, 1995.
- 37 Daniel A Spielman and Shang-Hua Teng. Spectral sparsification of graphs. *SIAM Journal on Computing*, 40(4):981–1025, 2011.
- 38 Salil P. Vadhan. Pseudorandomness. *Foundations and Trends in Theoretical Computer Science*, 7(1-3):1–336, 2012.
- 39 Rephael Wenger. Extremal graphs with no  $C_4$ 's,  $C_6$ 's, or  $C_{10}$ 's. *Journal of Combinatorial Theory, Series B*, 52(1):113–116, 1991.





# Proofs of Catalytic Space

Krzysztof Pietrzak<sup>1</sup>

Institute of Science and Technology Austria, Austria  
pietrzak@ist.ac.at

---

## Abstract

---

Proofs of space (PoS) [Dziembowski et al., CRYPTO’15] are proof systems where a prover can convince a verifier that he “wastes” disk space. PoS were introduced as a more ecological and economical replacement for proofs of work which are currently used to secure blockchains like Bitcoin. In this work we investigate extensions of PoS which allow the prover to embed useful data into the dedicated space, which later can be recovered.

Our first contribution is a **security proof** for the original PoS from CRYPTO’15 in the random oracle model (the original proof only applied to a restricted class of adversaries which can store a subset of the data an honest prover would store). When this PoS is instantiated with recent constructions of maximally depth robust graphs, our proof implies basically optimal security.

As a second contribution we show three different extensions of this PoS where useful data can be embedded into the space required by the prover. Our security proof for the PoS extends (non-trivially) to these constructions. We discuss how some of these variants can be used as **proofs of catalytic space** (PoCS), a notion we put forward in this work, and which basically is a PoS where most of the space required by the prover can be used to backup useful data. Finally we discuss how one of the extensions is a candidate construction for a **proof of replication** (PoR), a proof system recently suggested in the Filecoin whitepaper.

**2012 ACM Subject Classification** Theory of computation → Interactive proof systems

**Keywords and phrases** Proofs of Space, Proofs of Replication, Blockchains

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2019.59

**Related Version** A full version of the paper is available at [29], <https://eprint.iacr.org/2018/194.pdf>.

## 1 Introduction

### 1.1 Proofs of Space (PoS)

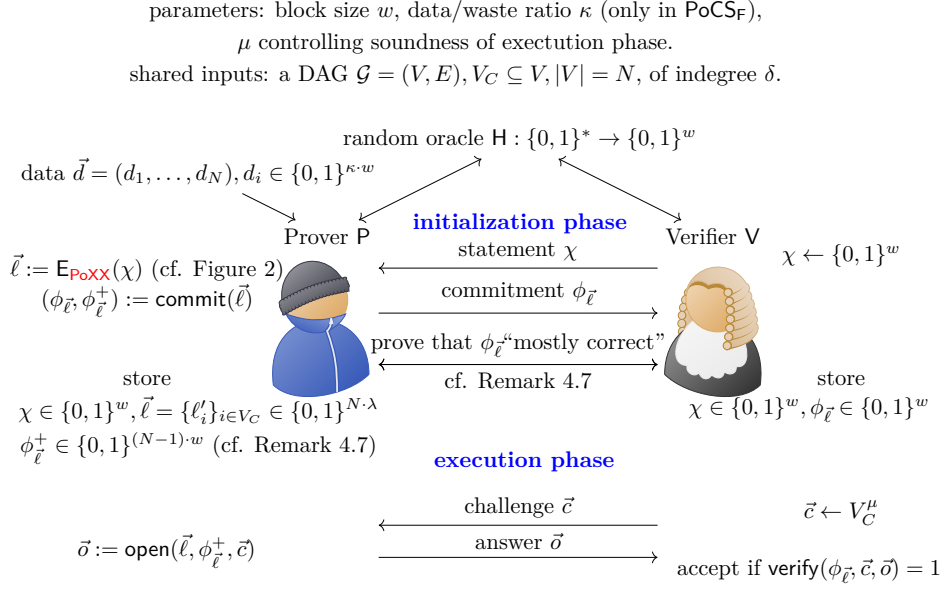
A proof of space (PoS) [16, 30, 3] is an interactive proof system in which a prover  $P$  can convince a verifier  $V$  that it “wastes” a large amount of disk-space. PoS were suggested as an alternative to proofs of work (PoW), which are currently used for securing blockchains including Bitcoin and Ethereum. PoS-based proposals include Spacemint [27] and the Chia network [1]. In the full version of this paper [29] we provide more discussion on sustainable blockchains and PoS.

The core of the pebbling-based PoS [16, 30] is a mode of operation for hash-functions  $E_{\text{PoS}}$ , which is specified by a directed acyclic graph (DAG)  $\mathcal{G} = (V, E)$  with a dedicated set  $V_C \subseteq V$  of  $|V_C| = N$  “challenge nodes”. The constructions in [16, 30] mostly differ in

---

<sup>1</sup> 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 682815/TOCNeT).





■ **Figure 1** Illustration of protocol structure of the proof of space  $\text{PoS}_o$ , our proofs of catalytic space  $\text{PoCS}_F$ ,  $\text{PoCS}_\phi$  and the proof of replication  $\text{PoR}$  (replace  $\text{PoXX}$  in the figure with any of those). In  $\text{PoCS}_F$   $\kappa$  is a parameter, in  $\text{PoCS}_\phi$ ,  $\text{PoR}$  set  $\kappa = 1$  and for  $\text{PoS}_o$  set  $\kappa = 0$  (i.e.,  $\vec{d}$  is empty) in the figure. The label size  $\lambda$  is  $w(\kappa + 1)$  in  $\text{PoCS}_F$  and  $w$  in  $\text{PoS}_o$ ,  $\text{PoR}$  and  $\text{PoCS}_\phi$ .

what type of graphs are used. The only input  $E_{\text{PoS}_o}$  takes is a short statement  $\chi$  which is used to sample a hash function  $H_\chi$  (modelled as random oracle in all our proofs), and it outputs a large file  $\vec{\ell} = \{\ell_i\}_{i \in V_C}$  which P must store. P sends a commitment  $\phi_{\vec{\ell}}$  to  $\vec{\ell}$  to V. To check the prover really stores this file, the verifier can occasionally send a random challenge  $i \in V_C$  to the prover, who then must open the label  $\ell_i \in \vec{\ell}$  of this file. If such audits happen sufficiently often, the rational thing for P to do is to store  $\vec{\ell}$ , and not recompute labels as they are requested. The high level proof structure of this PoS, denoted  $\text{PoS}_o$ , is illustrated in Figure 1, the underlying mode of operation, denoted  $E_{\text{PoS}_o}$ , is illustrated in Figure 2.

## 1.2 An Unconditional Security Proof for the [16] PoS

Informally, the security we want from a PoS is as follows: if a malicious prover  $\tilde{P}$  dedicates slightly less space than the honest prover would after the initialization phase, say  $(1 - \epsilon) \cdot N$  instead  $N$  for some small  $\epsilon > 0$ , then it should be “expensive” for him to pass the audit. Note that  $\tilde{P}$  can always pass the audit by simply recomputing the entire  $\vec{\ell}$  right before the audit, so the best we can hope for is that passing the audit is almost as expensive for  $\tilde{P}$  as it is to compute the entire  $\vec{\ell}$ .

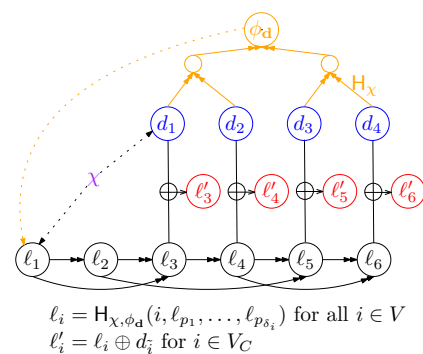
The first contribution in this paper is a security proof that shows  $\text{PoS}_o$  is a secure PoS in the random oracle model (Corollary 8 in §7.2). The existing proof from [16] only showed security against restricted adversaries who store a subset of the data  $\vec{\ell}$  an honest prover would store, but didn’t imply anything against more general adversaries who can store an arbitrary function of this data. We discuss this in more detail in §5.

When we instantiate  $E_{\text{PoS}_o}$  with recent constructions of depth-robust graphs, the security we get is basically optimal. Informally, for any  $\epsilon > 0$ , we can chose parameters such that any cheating prover who dedicates only an  $1 - \alpha$  fraction of the required space will fail to *efficiently* answer an  $\alpha - \epsilon$  fraction of the challenges (which simply ask to open some

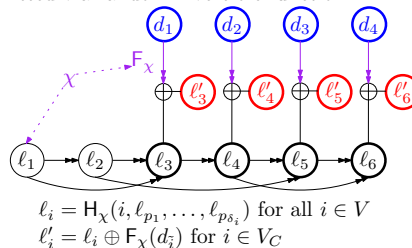
$E_{\text{PoS}_\circ}$ : The **proof of space** from [DFKP15] instantiated with (a toy example of) a depth-robust graph.



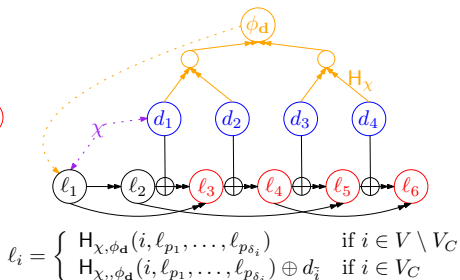
$E_{\text{PoCS}_\phi}$ : Our **proof of catalytic space** where the data is committed by a standard Merkle tree commitment  $\phi_d$ . The hash function for this commitment depends on  $\chi$ , the hash function for the labelling also on  $\phi_d$ .



$E_{\text{PoCS}_F}$ : Our **efficiently updatable proof of catalytic space** where the catalytic data committed via random invertible function  $F$ .



$E_{\text{PoR}}$ : Our **proof of replication** is similar to  $E_{\text{PoCS}_\phi}$ , but the data is XOR'ed to the labels as the computation goes on, not just at the end.



■ **Figure 2** Illustration of the graph based modes of operation used in the proof of space  $\text{PoS}_\circ$ , proof of catalytic space  $\text{PoCS}_\phi$  and its efficiently updatable variant  $\text{PoCS}_F$  and the proof of replication  $\text{PoR}$ . We use a toy example of a depth robust DAG  $\mathcal{G} = (V, E)$ ,  $V = \{1, \dots, 6\}$  with  $V_C = \{3, \dots, 6\}$  being the challenge nodes. The embedded data is shown in blue, the labels the prover stores are in red. The values represented by all nodes are in  $\{0, 1\}^w$ , except the bold nodes in  $\text{PoCS}_F$ , where the  $d_i$  are in  $\{0, 1\}^{w \cdot \kappa}$  and the  $l_i, l'_i, i \in V_C$  are in  $\{0, 1\}^{w \cdot (\kappa+1)}$ .

blocks in the file  $\vec{\ell}$  the prover is committed to). Thus, if say  $\alpha = 2\epsilon$ , the prover fails on an  $\epsilon$  fraction, and we can amplify this to be overwhelmingly close to 1 by using  $O(1/\epsilon)$  challenges in parallel. Above, with “efficiently recover”, we mean it needs parallel time<sup>2</sup>  $N$ , this is basically optimal in terms of time complexity, as running the entire initialization phase takes only (sequential) time  $4N$ . We will discuss how our proof compares with existing results in more detail in §2.

The efficiency of our schemes (i.e., proof size, proof generation time, proof verification time) are all in  $O(\log N)$ , where the hidden constant depends on the above mentioned  $\epsilon$  (i.e., the constant grows as  $\epsilon$  goes to 0).

### 1.3 Embedding Useful Data into a PoS

The file  $\vec{\ell} := E_{\text{PoS}_\circ}(\chi)$  the prover is supposed to store just wastes disk space, and cannot be used for anything useful. This makes sense, as after all a PoS is supposed to prove the dedicated space is wasted.

In this paper we investigate the setting where the space dedicated towards the PoS can at the same time be used to encode some useful data  $\vec{d}$ . We identify two applications for such objects, *proofs of replication* (PoR), which were (informally) introduced in the Filecoin

<sup>2</sup> Our proof is in the random oracle model, and parallel time  $N$  means  $N$  rounds of queries, where in each round one can make many queries in parallel. In sequential time just one query is allowed.

paper [23] and *proofs of catalytic space* (PoCS), which we introduce and motivate in this work. The naming of the latter is inspired by *catalytic space computations* [9, 10], which are computations that can be done in small space, but only if one is additionally given “catalytic space”. This space is initially filled with arbitrary (potentially incompressible) data, and must be in the same state after the computation finishes. It thus functions like a catalyst in chemical reactions.

We introduce three new proof systems which allow for such embedded data. Two of them are intended to be used as PoCS, denoted  $\text{PoCS}_\phi$  and  $\text{PoCS}_F$ . The  $\text{PoCS}_F$  scheme has a worse *rate* than  $\text{PoCS}_\phi$ , by which we mean the ratio  $\|\vec{d}\|/\|\vec{\ell}\|$  of embedded data vs. dedicated space, but unlike  $\text{PoCS}_\phi$ , it allows for efficient updates of the embedded data. The third scheme we introduce is called PoR and is intended to be used as a PoR. Our new proof systems are derived from the [16] PoS  $\text{PoS}_o$  by replacing its underlying mode  $E_{\text{PoS}_o}$  by another mode of operation  $E_{\text{PoXX}} \in \{E_{\text{PoCS}_\phi}, E_{\text{PoCS}_F}, E_{\text{PoR}}\}$ . These modes take as input  $\chi$  (just like  $E_{\text{PoS}_o}$ ), and additionally some data  $\vec{d} = \{d_i\}_{i \in V_C}$  and output a file  $\vec{\ell} := E_{\text{PoXX}}(\chi, \vec{d})$  to be stored. The data  $\vec{d}$  can be recovered from  $\vec{\ell}$  at any time. These four modes are all illustrated in Figure 2.

## 1.4 Fisch’s PoR

That depth-robust graphs are useful to construct proofs of replication has been observed independently by Ben Fisch, he discusses this in a BPASE’18 talk<sup>3</sup> which took place almost a month before this paper was posted in a public archive.

In a recent paper [18] Fisch starts developing the foundations of PoR. His paper addresses many conceptual and technical aspects of PoR, including PoR constructions based on depth-robust graphs similar to the ones in this paper. His security proofs crucially use the main technical from this paper, i.e., that pebbling lower bounds for parallel time complexity imply lower bounds in the random oracle model as stated in Theorem 7. In a subsequent work Fisch presents PoR based on depth robust graphs with even better provable security guarantees [19]. The construction in that paper is conceptually somewhat different from ours in terms of the underlying graphs but also in the way in which the data is embedded. The graphs are stacked depth robust graphs, and are somewhat reminiscent of the simple and elegant PoS of Ren and Devadas [30] which is based on stacked expanders. We’ll say more on the efficiency of those constructions in §2.

## 1.5 Properties of PoS, PoCS and PoR

Fisch [18] observes that for any meaningful definition of PoR, a *PoR necessarily is also a PoS* as defined in [16]. Also a PoCS must necessarily be a PoS.

We (non-trivially) extend our security proof for  $\text{PoS}_o$  to prove that also the schemes  $\text{PoCS}_\phi$ ,  $\text{PoCS}_F$ , PoR are PoS (The final bound for  $\text{PoS}_o$ ,  $\text{PoCS}_\phi$ , PoR is stated in Corollary 8 in §8.2, the bound for  $\text{PoCS}_F$  is in the full version [29]. On the other hand, we observe that *being a PoS with the option to embed useful data is not sufficient to constitute a good PoCS or PoR*. Moreover the “whish list” of properties one might have for PoCS and PoR is somewhat contradictory. Our PoR candidate PoR is not a good PoCS, while  $\text{PoCS}_\phi$ ,  $\text{PoCS}_F$  do not make for a good PoR, as we’ll elaborate next.

The most important property we want from a PoCS is that any particular block of data from  $\vec{d} = \{d_i\}_{i \in V_C}$  cannot be recovered too efficiently from  $\vec{\ell}$ . The reason is that otherwise the PoCS wouldn’t compose: a malicious prover  $\tilde{P}$  could run a PoCS for some statement  $\chi$ ,

<sup>3</sup> [https://www.youtube.com/watch?v=8\\_90NpyRZEI](https://www.youtube.com/watch?v=8_90NpyRZEI)

and at the same time using the embedded data  $\vec{d}$  for a PoCS for other statement  $\chi'$ , thus pretending to dedicate more space than it does. To prevent this, we want the PoCS to *lock* the catalytic data, by which we mean accessing any particular block  $d_i \in \vec{d}$  should be almost as expensive as recovering the entire  $\vec{d}$  from  $\vec{\ell}$ .

In a typical application of a PoR, the data  $\vec{d}$  is not chosen by P but provided by V, together with some replication parameter  $r \in \mathbb{N}$  (and statement  $\chi$ ). P will then run the PoR for various statements  $\chi_1, \dots, \chi_r$  (generated from  $\chi$ ), each embedding  $\vec{d}$ . Informally, the security property we want is that a prover who later successfully passes the audits must have stored  $r$  *redundant* copies of  $\vec{d}$  (as  $\vec{d}$  can be incompressible, the redundancy requirement implies that PoR is a PoS). So unlike for a PoCS, in a PoR, being able to recover any data block efficiently is actually a feature, not an issue that breaks security.

We'll discuss those properties and how our schemes do or don't achieve them in more detail in §4, after having defined our various modes. We'll keep the discussion about the exact notion of a PoCS or PoR, and in particular any properties beyond being a PoS, like the locking and replication property mentioned above, informal. We expect future work to come up with the right definition for a PoCS, and also [18] is probably also not the last word on definitional issues for PoRs. There certainly are more properties one might need from a PoR or PoCS in particular applications that have not yet been identified, coming up with the right definitions and constructions satisfying them (in particular, showing that the constructions presented in this work do or do not satisfy them) is a promising research agenda.

## 2 Comparison With Previous Work

We somewhat divert from [16] when formally defining the security as a PoS. In [16], a PoS is defined to be  $(N_0, N_1, T)$ -secure if an adversary who stores a file  $\vec{\ell}$  of size  $N_0$  (recall that we measure size in blocks, typically of size something like  $w = 256$  bits) after the initialization phase, uses  $N_1$  space and  $T$  time during the proof executing phase, will fail in making the verifier accept with overwhelming probability. Below we shortly compare the four pebbling-based instantiations of the PoS<sub>o</sub> construction that so far have been suggested, and what security has been proven for them. Those just differ in the graphs  $\mathcal{G} = (V, E)$  and the dedicated set of challenge vertices  $V_C \subseteq V, |V_C| = N$ . We also mention the total number of edges  $|E|$ , as this basically determines the efficiency of the initialization procedure, and the indegree  $\delta$ , as this determines the size of the proofs and also the time to generate and verify proof. Let us mention that there is one work [3] constructing PoS using a completely different approach than graph-pebbling, for space reasons we'll only discuss his in the full version [29].

### 2.1 The original PoS [16]

[16] introduced the notion of PoS and gave two constructions, the first is

$$(\Theta(N/\log(N)), N/\log(N), \infty)\text{-secure with } |V| = N, \delta = 2, |E| = 2N$$

and based on a graph with high space pebbling complexity by Paul, Tarjan and Celoni [28], the second uses a rather sophisticated construction combining random bipartite graphs, superconcentrators and depth-robust graphs [17] and is

$$(\Theta(N), \infty, \Theta(N))\text{-secure with } |V| = N, \delta \in O(\log \log N), |E| \in O(N \log \log N)$$

## 2.2 The Ren-Devadas PoS [30]

Ren and Devadas [30] propose a very elegant instantiation of PoS<sub>o</sub> using stacked expanders and give a proof for it which in terms of security improves upon both constructions from [16] (just  $|E|$  is asymptotically larger). For any  $\alpha \in [0, 0.5]$ , their proof implies security (almost)

$$(\alpha \cdot N, (1 - \alpha) \cdot N, \infty)\text{-secure with } |V| \in O(N \log N), \delta \in O(1), |E| \in O(N \log N)$$

For say  $\alpha = 1/3$ , this means an adversary storing  $N_0 = N/3$  blocks after initialization, must use at least  $N_1 = 2N/3$  space during execution. Their construction is a stack of  $\log(N)$  expanders of indegree 2, and  $V_C$  is the graph on top of this stack.

## 2.3 Fisch’s tight PoS [19]

The PoS underlying Fisch’s recent PoR construction which we shortly discussed in §1.4, is based on a PoS that *for any*  $\epsilon > 0$  achieves

$$(N \cdot (1 - \epsilon), \infty, N)\text{-security with } |V| = O(N \log(1/\epsilon)), \delta \in O(1), |E| \in O(N \log(1/\epsilon))$$

Fisch provides concrete bounds for all the constants, the bounds are so good that he gets a very practical construction for parameters where the proofs guarantee good practical security. For our construction, which we discuss below, this is not the case. As there’s a huge gap between the lower and upper bounds on the security we can prove for PoS based on the simple depth robust graphs from [4, 6], it’s not clear whether the actual security of the “stacked” constructions as used in [19, 30] really is practically better, or if those constructions just allow for much tighter proofs, while not actually having better security in practice. Settling this is an interesting open question.

## 2.4 Our PoS

In this work we use the depth-robust graphs from [4, 6] to instantiate PoS<sub>o</sub>, and also our three new constructions which allow to embed useful data. *For any*  $\epsilon > 0$ , we can instantiate it as to get

$$(N \cdot (1 - \epsilon), \infty, N)\text{-security with } |V| = 4N, \delta \in O(\log(N)), |E| \in O(N \log N)$$

This might not seem terribly impressive, note that unlike [30] we don’t claim any lower bound on  $N_1$ , the space a cheating adversary must dedicate during proof execution. And asymptotically,  $|E|$  is larger than in the second construction of [16] which (ignoring constants) has the same security. But as we’ll explain next, we improve upon all existing constructions, except the subsequent work by Fisch [19], in three crucial points.

- 1. Unconditional Proof:** Our proof holds unconditionally (in the random oracle model), whereas [16, 30] only argued security against restricted adversaries who store a subset of the file an honest prover would store. Let us mention that for such relaxed adversaries, we can also prove bounds on the space a successful prover needs during execution.<sup>4</sup>

---

<sup>4</sup> Basically, for this restricted class of adversaries, whatever bound on time and/or space is proven for the underlying graph translates to a time and/or space bound for the construction. Our unconditional proof only translates parallel time complexity. The graphs we use to instantiate our construction are depth-robust, and such graphs are known [5] to have high “cumulative pebbling complexity”, which (for restricted adversaries as just mentioned) translates to the fact that if adversary runs in  $T$  rounds during proof execution, it must use  $\Omega(N^2/T)$  space on average during this computation.

2. **Tight Bound:** We get tight security: for any constant  $\epsilon > 0$ , we can instantiate our PoS to be  $((1 - \epsilon) \cdot N, \infty, N)$  secure. Equivalently, an adversary storing just an  $\epsilon$  fraction less than the honest prover, and which can run in time  $T = N$ , will still fail to make the verifier accept with overwhelming probability.

Having such a tight bound is crucial for many applications, as it means we get security against an adversary dedicating an  $(1 - \epsilon)$  fraction of the space for any  $\epsilon > 0$ . Note that even the (proof of the) [30] construction doesn't imply any security against adversary who dedicates just  $N_0 = N/2$ , i.e., half the claimed space.

3. **Security Against Parallelism:** The security we prove even holds if we strengthen the meaning of the parameter  $T$  from “total number of oracle queries”, to “total number of *parallel* oracle queries”, where each parallel query can contains many inputs, as long as in total they are bound by an exponential.

This stronger security notion implies that even massive parallelism doesn't help a potential adversary. This is useful in a setting where the timepoint at which audits happen is not known to the prover (in proofs of replication this can be achieved), and we have a bound on the latency of network between prover and verifier (so the prover cannot make  $T = N$  *sequential* computations in time less than this latency). Here we can be sure a prover who passes the audits really dedicates the claimed space, and does not simply reinitialize the entire space once the audit starts fast enough using massive parallelism. Compare this to the construction from [30], which can be initialized in sequential time  $\log(N)$  using parallelism  $N$ .

We will not use the formalism from [16] to quantify security outside of this subsection, but in our security statements explicitly state what is achieved, which should be easier to parse. The PoS security of  $E_{\text{PoS}_0}$  is stated in Corollary 8, The PoS security of  $E_{\text{PoCS}_\phi}$  and  $E_{\text{PoR}}$  in Corollary 13 and the PoS security of  $E_{\text{PoCS}_F}$  in the full version [29].

### 3 Basic Notation and Definitions

#### 3.1 Notation

For an object  $X$ ,  $\|X\|$  denotes its bitlength, for a set  $\vec{x}$ ,  $|\vec{x}|$  is the number of elements in  $\vec{x}$ . For an integer  $m$  we denote  $[m] \stackrel{\text{def}}{=} \{1, 2, \dots, m\}$  and for  $a, b \in \mathbb{R}$  we denote  $[a, b] \stackrel{\text{def}}{=} \{c : a \leq c \leq b\}$ . With  $\{0, 1\}^{\leq m}$  we denote the set of strings of length  $\leq m$ .

We typically use small greek letters  $\iota, \delta, \omega, \mu, \nu, \epsilon, \dots$  for our parameters used to quantify security, efficiency etc.. An exception is  $N$  which throughout denotes the space requirement of a prover. All these parameters are values in  $\mathbb{N}$  except  $\epsilon$  which is in  $[0, 1]$ . For the security games considered in this paper we use capital greek letters  $\Phi, \Lambda$ . The sans-serif font is used for interactive systems like parties  $V, P, \tilde{P}, A$  (modelled as randomized interactive Turing machines), functions  $H, F, g, f$  or algorithms like commitments discussed below ( $V, P$  and  $\tilde{P}$  are reserved for an honest verifier, an honest prover, and a potentially malicious prover, respectively). We use bold letters  $\vec{\ell}, \vec{d}, \vec{o}, \vec{c}, \dots$  for sets (usually ordered) of values, except for graph notation where we use simply  $\mathcal{G} = (V, E)$  to denote a graph with vertices  $V$  and directed edges  $E$ .

We will often consider a subset  $V_C \subset \mathbb{N}, |V_C| = N$  of challenge nodes. It will be convenient to define concise notion for mapping  $V_C$  to  $[N]$ , which we do using a tilde, i.e.,

$$V_C = (v_1, \dots, v_N) \quad \Rightarrow \quad (\tilde{v}_1, \dots, \tilde{v}_N) = (1, \dots, N) \quad (1)$$



## 3.2 Random Oracles

### 3.2.1 Fresh Random Oracles.

If  $H$  is a fixed random oracle and  $z \in \{0, 1\}^*$ , we denote with  $H_z$  the function  $H_z(\cdot) = H(z, \cdot)$ . If  $z$  is random and long enough (concretely, the amount of non-uniform advice an adversary has on  $H$  is not too large exponential in  $\|z\|$ ), we can treat  $H_z$  as a fresh uniformly random oracle [14]. We do this repeatedly in this work without always explicitly mentioning it.

### 3.2.2 The Parallel Random Oracle Model.

We prove security of our schemes in the *parallel random oracle model*, where in every round an adversary can output a set  $x_1, \dots, x_i$  of queries, and it receives the outputs  $H(x_1), \dots, H(x_i)$  at the beginning of the next round. For us the number of rounds will be important, but the total number of queries is secondary. Although also the total number of queries must be bound, in our proofs it can be as large as exponential in the block size  $w$ , and for the basic PoS<sub>o</sub>, the number of queries during the initialization phase can be even unbounded (not so for the other schemes). We will denote the number of oracle queries to  $H$  an adversary is allowed to make in the initialization and proof execution phase by  $q_1^H$  and  $q_2^H$ , respectively.

## 3.3 Commitments

We will make extensive use of a Merkle-tree commitment scheme, which allows to compute a short commitment to a long string, and later efficiently open any particular location of that long string. It is specified by a triple of algorithms `commit`, `open`, `verify` which use a hash-function  $H : \{0, 1\}^* \rightarrow \{0, 1\}^w$  as a building block. For the security as a commitment scheme, it's sufficient for  $H$  to be collision resistant. In our proofs we will sometimes need to extract committed values from the committing party, for this we must assume  $H$  is given as an oracle so our reduction can observe all queries.

If it's relevant what hash function is used it's shown as superscript (otherwise one can just assume any collision-resistant hash function is used). A party  $A$  which wants to commit to values  $\vec{x} = (x_1, \dots, x_m)$  invokes

$$(\phi_{\vec{x}}, \phi_{\vec{x}}^+) := \text{commit}^H(\vec{x})$$

here  $\phi_{\vec{x}}^+ \in \{0, 1\}^{(m-1)w}$  denotes the values of all inner nodes in the Merkle-tree, which are required to later efficiently open any position in  $\vec{x}$ , and  $\phi_{\vec{x}} \in \{0, 1\}^w$  is the value at the root, which is the commitment.

Once  $A$  announces  $\phi_{\vec{x}}$  it is committed to  $\vec{x}$ . It can then open any subset  $\vec{i} \subseteq [m]$  of the committed values (i.e.,  $\{x_i\}_{i \in \vec{i}}$ ) by invoking

$$\vec{o} := \text{open}^H(\vec{x}, \phi_{\vec{x}}^+, \vec{i}) \in \{0, 1\}^{\leq |\vec{i}| \lceil \log(m) \rceil \cdot w}.$$

Everyone can verify that  $\vec{o}$  is the correct opening by invoking  $\text{verify}^H(\phi_{\vec{x}}, \vec{i}, \vec{o})$  and accepting iff this value is 1.

## 3.4 Random Strings are Incompressible

In our proofs we'll repeatedly use the following fact, which states that a random string cannot be compressed.

► **Fact 1** (statement from [13]). *For any randomized encoding procedure  $\text{enc} : \{0, 1\}^r \times \{0, 1\}^n \rightarrow \{0, 1\}^m$  and decoding procedure  $\text{dec} : \{0, 1\}^r \times \{0, 1\}^m \rightarrow \{0, 1\}^n$  where*

$$\Pr_{x \leftarrow \{0, 1\}^n, \rho \leftarrow \{0, 1\}^r} [\text{dec}(\rho, \text{enc}(\rho, x)) = x] \geq \delta$$

we have  $m \geq n - \log(1/\delta)$ .

## 4 Overview of Our Modes and Protocols

In this section we will formally define the mode  $\mathbf{E}_{\text{PoS}_o}$  underlying the PoS from [16] and our new modes  $\mathbf{E}_{\text{PoR}}, \mathbf{E}_{\text{PoCS}_\phi}, \mathbf{E}_{\text{PoCS}_F}$  as illustrated in Figure 2. The actual protocols using those modes as illustrated in Figure 1 will then be defined in §4.5. As we define the modes, we will also continue our discussion from §1 showing how they (fail to) perform as PoCS or PoR. In particular, we'll show that our mode  $\mathbf{E}_{\text{PoR}}$  is not locking (and thus not suitable as PoCS), whereas  $\mathbf{E}_{\text{PoCS}_\phi}, \mathbf{E}_{\text{PoCS}_F}$  do not imply replication. All the modes are defined over a DAG  $\mathcal{G} = (V, E)$ , for  $i \in V$  we denote with  $\text{parents}(i) = \{j : (j, i) \in E\}$  the parents of  $i$ , and we define  $\vec{\ell}_{\text{parents}(i)} = \{\ell_j : j \in \text{parents}(i)\}$ .

### 4.1 The Mode $\mathbf{E}_{\text{PoS}_o}$

In the basic PoS the file  $\vec{\ell} := \mathbf{E}_{\text{PoS}_o}(\chi)$  contains the “labels” of nodes in  $V_C$ , where the labels of the nodes of the underlying DAG  $\mathcal{G} = (V, E)$  are computed in topological order by hashing (using a fresh random oracle sampled using  $\chi$ ) the labels of its parents

$$\forall i \in V : \ell_i = \mathbf{H}_\chi(i, \vec{\ell}_{\text{parents}(i)}) \quad , \quad \vec{\ell} \stackrel{\text{def}}{=} \{\ell_i\}_{i \in V_C} \quad (\mathbf{E}_{\text{PoS}_o}) \quad (2)$$

The most obvious way to somehow embed data  $\vec{d} = \{d_i\}_{i \in [N]}$  into this basic PoS is to simply XOR the data blocks to the labels in  $V_C$ . There are two natural ways to do this, XOR the data to the labels as the computation goes on, or first compute the labels and then XOR the data to it. The first approach is basically what we do in our construction PoR, and the second in  $\text{PoCS}_\phi$ . Before computing the labels, we first commit to  $\vec{d}$ , and then sample a fresh random oracle to compute the labels using this commitment. We'll explain below why without this trick our constructions would miserably fail to be PoS.

### 4.2 The Mode $\mathbf{E}_{\text{PoR}}$

In our PoR  $\vec{\ell} := \mathbf{E}_{\text{PoR}}(\chi, \vec{d})$ , the data  $\vec{d} = \{d_i\}_{i \in [N]}$  is first committed

$$(\phi_{\vec{d}}, \phi_{\vec{d}}^+) := \text{commit}^{\mathbf{H}_\chi}(\vec{d}) .$$

Then  $\phi_{\vec{d}}$  is used to sample a fresh random oracle  $\mathbf{H}_{\chi, \phi_{\vec{d}}}(\cdot) = \mathbf{H}_\chi(\phi_{\vec{d}}, \cdot)$ , which is then used to compute the labels. For labels of a node  $i \in V_C$ , one additionally XORs the data block  $d_i$  (recall that  $\{\vec{i}\}_{i \in V_C} = [N]$ ) to the label right after it has been computed.

$$\ell_i = \begin{cases} \mathbf{H}_{\chi, \phi_{\vec{d}}}(i, \vec{\ell}_{\text{parents}(i)}) & \text{if } i \in V \setminus V_C \\ \mathbf{H}_{\chi, \phi_{\vec{d}}}(i, \vec{\ell}_{\text{parents}(i)}) \oplus d_i & \text{if } i \in V_C \end{cases} \quad , \quad \vec{\ell} = \{\ell_i\}_{i \in V_C} \quad (\mathbf{E}_{\text{PoR}}) \quad (3)$$

### 4.3 The Mode $\mathbf{E}_{\text{PoCS}_\phi}$

In our PoCS  $\vec{\ell} := \mathbf{E}_{\text{PoCS}_\phi}(\chi, \vec{d})$  one first computes  $\phi_{\vec{d}}$  as above, uses this to sample a fresh random oracle  $\mathbf{H}_{\chi, \phi_{\vec{d}}}$  to compute labels (as in the basic PoS<sub>o</sub>), and only then XORs the data to the labels to be stored

$$\forall i \in V : \ell_i = \mathbf{H}_{\chi, \phi_{\vec{d}}}(i, \vec{\ell}_{\text{parents}(i)}) \quad (4)$$

$$\forall i \in V_C : \ell'_i = \ell_i \oplus d_i, \quad \vec{\ell} = \{\ell'_i\}_{i \in V_C} \quad (\mathbf{E}_{\text{PoCS}_\phi}) \quad (5)$$

As mentioned above, the fact that the random oracle  $\mathbf{H}_{\chi, \phi_{\vec{d}}}$  used to compute the labels depends on the commitment  $\phi_{\vec{d}}$  to  $\vec{d}$  is crucial, let us sketch why. Assume we'd change  $\mathbf{H}_{\chi, \phi_{\vec{d}}}$  to  $\mathbf{H}_\chi$  in the definition of  $\mathbf{E}_{\text{PoCS}_\phi}$ . Now a malicious prover (in the protocol  $\text{PoCS}_\phi$  to be defined in §4.5) could set the  $\ell'_i$  to be stored to whatever it wants (and thus also store them using low space). Only after choosing the  $\ell'_i$ , it then fixes the data  $\vec{d} = \{d_i\}_{i \in [N]}$  by “equivocating” it, i.e., setting it as  $d_i := \ell'_i \oplus \ell_i$ , so everything is consistent. This malicious behaviour (not using any space) cannot be distinguished from honest behaviour, thus it's not a PoS.

Now let us observe that  $\text{PoCS}_\phi$  is not a good PoR, as it doesn't imply replication. A prover who is supposed to compute and store  $\vec{\ell}_i := \text{PoCS}_\phi(\chi_i, \vec{d})$  for  $r$  statements  $\chi_1, \dots, \chi_r$  but the same  $\vec{d}$  can instead store  $\vec{d}$  once in the clear, and for then for each  $\chi_j$  only store the labels  $\{\ell_i\}_{i \in V_C}$  as in eq.(4) instead  $\{\ell'_i = \ell_i \oplus d_i\}_{i \in V_C}$ , i.e., avoid the XORing step of eq.(5). Note that this prover has only stored one copy of  $\vec{d}$ , instead of storing it  $r$  times redundantly, while it can still pass the audits for all  $\chi_i, i \in [r]$  because it can compute the correct labels  $\ell'_i$  using its single copy of  $\vec{d}$  as  $\ell'_i = \ell_i \oplus d_i$ . The prover here doesn't seem to gain much, in particular it doesn't save on space by deviating from the honest behaviour. But in a PoR we probably want to enforce replication, or at least argue that it's not rational for a prover to deviate, and there are settings where deviating as just explained can be rational. Assume the prover has large remote storage space, but only low bandwidth to access it. By deviating as explained, it can use the space for the  $r$  proofs without large communication, in particular, without ever having to send  $\vec{d}$  to the remote disk.

In the other direction one can argue that  $\mathbf{E}_{\text{PoR}}$  is not a good PoCS as given all labels  $\{\ell_i\}_{i \in V}$  as in eq.(3), one can efficiently recover the embedded data as  $d_i = \ell_i \oplus \mathbf{H}_{\chi, \phi_{\vec{d}}}(i, \vec{\ell}_{\text{parents}(i)})$ , so it doesn't provide the locking property we want from a PoCS. The above argument highlights a problem with the PoCS security of  $\mathbf{E}_{\text{PoR}}$ , but is not totally convincing, as the prover actually only needs to store the  $\ell_i$  for  $i \in V_C$  (not all  $i \in V$ ), so it couldn't necessarily recover those labels efficiently.

### 4.4 The Mode $\mathbf{E}_{\text{PoCS}_F}$

Besides  $\text{PoCS}_\phi$ , we propose a second PoCS  $\text{PoCS}_F$ , which allows for efficient updates. Instead of committing to  $\vec{d}$  and using this commitment to sample the random oracle  $\mathbf{H}_{\chi, \phi_{\vec{d}}}$  for computing the labels as in  $\text{PoCS}_\phi$ , in  $\text{PoCS}_F$  the labels are computed directly using  $\mathbf{H}_\chi$ , i.e., independently of  $\vec{d}$ . To prevent the “equivocation” attack outlined above, in  $\text{PoCS}_F$  the prover samples (using  $\chi$ ) a random invertible function  $\mathbf{F}_\chi : \{0, 1\}^{(\kappa+1) \cdot w} \rightarrow \{0, 1\}^{\kappa \cdot w}$  and applies it to the data before XORing it to the label, where  $\kappa$  is a parameter discussed below. The labels  $\ell_i, i \in V \setminus V_C$  have length  $w$  bits, the labels  $\ell_i, i \in V_C$  are  $(\kappa + 1) \cdot w$  bits long. Below  $\mathbf{H}_\chi : \{0, 1\}^{\leq \iota} \rightarrow \{0, 1\}^{(\kappa+1) \cdot w}$ , and  $\mathbf{H}_\chi(\cdot)|_w$  means we cap the output after  $w$  bits.

$$\ell_i = \begin{cases} \mathbf{H}_\chi(i, \vec{\ell}_{\text{parents}(i)}) & \text{if } i \in V_C \\ \mathbf{H}_\chi(i, \vec{\ell}_{\text{parents}(i)})|_w & \text{if } i \in V \setminus V_C \end{cases} \quad (6)$$

$$\forall i \in V_C : \ell'_i = \ell_i \oplus \mathbf{F}_\chi(d_i), \quad \vec{\ell} = \{\ell'_i\}_{i \in V_C} \quad (\mathbf{E}_{\text{PoCS}_\phi}) \quad (7)$$

A random invertible function  $F_\chi$  as used in this construction can be constructed efficiently, and with almost no loss in concrete security (which is crucial for our application) instantiated from a random oracle [21].

PoCS $_\phi$  has a better rate than PoCS $_F$ . In PoCS $_\phi$  we have rate  $\frac{\|\vec{d}\|}{\|\vec{\ell}\|} = 1$ , so the stored file  $\vec{\ell}$  is as big as the data  $\vec{d}$  that can be recovered from it.<sup>5</sup> The rate of PoCS $_F$  is only  $\frac{\|\vec{d}\|}{\|\vec{\ell}\|} = \frac{\kappa}{\kappa+1}$ . For efficiency reasons the  $\kappa$  can't be too large ( $\kappa \approx 10$  is reasonable, then the size of  $\vec{\ell}$  is  $\approx 10\%$  larger than  $\vec{d}$ ).

But unlike PoCS $_\phi$ , PoCS $_F$  allows for fast updates of the catalytic data: if P wants to change a single data block  $d_i$  (embedded in  $\ell'_i = \ell_i \oplus F_\chi(d_i)$ ) to  $d'_i$  then it can simply replace the label  $\ell'_i$  with  $\ell'_i \oplus F_\chi(d_i) \oplus F_\chi(d'_i)$ . It then must also update the commitment  $(\phi_\ell, \phi_\ell^\perp)$ , but this takes only time  $\log(N)$ . For this update P actually needs to know the currently embedded data block  $d_i$ . There are natural settings where  $\vec{d}$  is readily available in the clear. For example if the catalytic data  $\vec{d}$  encoded in  $\vec{\ell}$  is used as backup, and a working copy of  $\vec{d}$  is available in the clear. So we think this feature might be useful. This mode is somewhat different than the other three modes considered. Below we define and analyze the PoS $_o$ , PoR, PoCS $_\phi$  modes together as they're very similar, but PoCS $_F$  is different, and for space reasons the precise definitions and security proofs are only given in the full version [29].

► Remark (efficiently updatable PoR). Looking at Figure 2, one might wonder why there's no mode E $_{\text{PoR}_F}$ , where the data  $\vec{d}$  is first pre-processed by  $F_\chi$  as in E $_{\text{PoCS}_F}$ , but then XORed to the labels right after they are computed (as in E $_{\text{PoR}}$ ). The reason is that the only advantage of preprocessing  $\vec{d}$  using  $F_\chi$  as in E $_{\text{PoCS}_F}$  instead of committing to it as in E $_{\text{PoCS}_\phi}$  is the fact that it makes updating data blocks cheap. But if we XOR the data to the labels right after it has been computed as in E $_{\text{PoR}}$ , then updating a block in label  $\ell_i$ , will also change all subsequent labels  $\ell_j, j > i$ , even if the hash function used to compute labels is independent of  $\vec{d}$ . Thus, this update is not cheap after all. It's an interesting open problem to construct a candidate for a PoR where data blocks can be efficiently updated.

## 4.5 The Protocols PoS $_o$ , PoCS $_\phi$ and PoR

The protocols we consider in this work are a generalization of the pebbling-based PoS from [16] PoS $_o$ , where we allow the prover to chose and embed additional data  $\vec{d}$  into the file  $\vec{\ell}$  to be stored. We define PoS $_o$ , PoCS $_\phi$ , PoR together as they are very similar, we use PoXX  $\in \{\text{PoS}_o, \text{PoCS}_\phi, \text{PoR}\}$  as placeholder for any of those constructions.

$w, \mu$  : A block length  $w$  ( $w = 256$  is a typical value) and a statistical security parameter  $\mu$ .

$\mathcal{G}$  : A directed acyclic graph  $\mathcal{G} = (V, E)$  with max. indegree  $\delta$  and a designated set  $V_C \subseteq V$  of “challenge nodes” of size  $N = |V_C|$ .

$H$  : A hash function, which for the proof is modelled as a random oracle  $H : \{0, 1\}^{\leq \iota} \rightarrow \{0, 1\}^w$  which takes inputs of length at most  $\iota = (\delta + 2) \cdot w$  bits.

The space required by the honest prover is  $\approx N \cdot w = |V_C| \cdot w$  bits (we'll discuss the exact space requirement in Remark 4.7).

<sup>5</sup> The prover also must store opening information  $\phi_\ell^\perp$  for a Merkle commitment, but as we'll discuss in Remark 4.7 this is small compared to  $\vec{\ell}$ .

## 4.6 Initialization

V picks a random statement  $\chi \in \{0, 1\}^w$  and sends it to P.

If  $\text{PoXX} \neq \text{PoS}_o$ , the prover P can choose any data  $\vec{d} = \{d_i\}_{i \in [N]}$ ,  $d_i \in \{0, 1\}$  and then computes the file to store (if  $\text{PoXX} = \text{PoS}_o$ ,  $\vec{d}$  is empty)

$$\vec{\ell} := E_{\text{PoXX}}(\chi, \vec{d})$$

as shown in Figure 2 and explained in eq.(2)-(4).

Finally P computes the commitment  $(\phi_{\vec{\ell}}, \phi_{\vec{\ell}}^{\pm}) := \text{commit}(\vec{\ell})$  for all the labels  $i \in V_C$ , sends the short commitment  $\phi_{\vec{\ell}}$  to V and locally stores the opening information  $\phi_{\vec{\ell}}^{\pm}$ . This concludes the initialization if we assume P is honest during this phase (we'll discuss the general case in Remark 4.7).

At the end of the initialization phase the verifier stores the short strings  $\chi, \phi_{\vec{\ell}}$ . The prover stores  $\chi, \phi_{\vec{\ell}}^{\pm}$  and additionally a large file  $\vec{\ell} = \{\ell_i\}_{i \in V_C}$  of size  $\|\vec{\ell}\| = w \cdot N$ .

## 4.7 Proof execution

The protocol where  $P(\vec{\ell}, \chi, \phi_{\vec{\ell}}^{\pm})$  convinces  $V(\chi, \phi_{\vec{\ell}})$  that it stores  $\vec{\ell}$  is very simple. V samples a few nodes from the challenge set  $\vec{c} = (c_1, \dots, c_{\mu}) \subset V_C$  at random, and sends the challenge  $\vec{c}$  to P. P sends openings  $\vec{\sigma} := \text{open}(\vec{\ell}, \phi_{\vec{\ell}}^{\pm}, \vec{c})$  to the labels  $\{\ell_i\}_{i \in \vec{c}}$  to V, who then accepts iff  $\text{verify}(\phi_{\vec{\ell}}, \vec{c}, \vec{\sigma}) = 1$ .

► **Remark (prover is honest during initialization).** For most of the paper we will assume that even a malicious prover  $\tilde{P}$  follows the protocol during the initialization phase (i.e., behaves like the honest P). Of course we can't make this assumption in practice, that's why pebbling-based PoS have an extra communication round at the end of the initialization phase where – for some statistical security parameter  $\nu$  – V challenges  $\tilde{P}$  to open  $\nu$  labels and their parents to check that they were correctly computed. For this,  $\tilde{P}$  initially sends a commitment to all nodes  $V$ , not just  $V_C$ , and (except for  $\text{PoS}_o$ ) also a commitment to  $\vec{d}$ . If  $\tilde{P}$  committed to labels  $\{\ell_i^*\}_{i \in V}$  where it cheated on an  $\epsilon$  fraction of the labels, i.e., for  $\text{PoS}_o$  this means we have  $\ell_i^* \neq H_{\chi}(i, \ell_{p_1}^*, \dots, \ell_{p_{\delta_i}}^*)$ , then  $\tilde{P}$  will fail to pass this check with probability  $1 - (1 - \epsilon)^{\nu}$ .  $\tilde{P}$  can still get away with cheating at a small fraction of labels, but one can easily take care of this in the proof by assuming that storing such inconsistent labels can be done by P “for free”. As this is a minor technicality in the proof, we ignore this as not to obfuscate the main technical contributions.

► **Remark (P's space).** The size of the file  $\vec{\ell}$  is  $N \cdot w$  bits, which is basically the same as the  $(N - 1) \cdot w$  bits required to store the opening info  $\phi_{\vec{\ell}}^{\pm}$  of the Merkle-tree commitment to  $\vec{\ell}$ : on the 0'th level of the tree (the leaves) we have the  $N$  labels, then on level 1 we have  $N/2$  values, on level 2 we have  $N/4$  values, etc., for a total of  $N/2 + N/4 + \dots + 2 + 1 = N - 1$  labels of internal nodes that constitute  $\phi_{\vec{\ell}}^{\pm}$ . But the prover can decide to not store levels  $1 \dots k$ , thus saving only  $(N - 1)/2^k$  blocks, while all values in the omitted layers can be recomputed by hashing at most  $2^k$  leaf values (i.e., values from  $\vec{\ell}$ ). Thus, for say  $k = 5$ , the Merkle tree requires just a  $1/32$  fraction of the space of  $\vec{\ell}$ , but requires to hash 32 values. In practice that wouldn't be expensive as those leaf labels can always be stored consecutively on a disk, and thus reading them comes at small cost compared to reading the first random block (and hashing 32 blocks is not expensive compared to a disk access). In the discussions in this writeup we will thus mostly ignore the space required for storing the opening information  $\phi_{\vec{\ell}}^{\pm}$ .

## 5 The Main Proof Ideas

In this section we’ll discuss the main ideas used in the proofs of this paper, and give an overview of the work we borrowed ideas from. Pebbling is a game played on directed acyclic graphs (DAG), where a player can put pebbles on nodes of the graph according to some rules, and its goal is usually to pebble some particular node or set of nodes using as few “resources” as possible. There are various pebbling games one can consider, in this work we just consider the basic black-pebbling game, where the player can put a pebble on a node if all of its parents have pebbles. The resource considered is typically time (i.e., how many rounds it takes) or space (i.e., the maximum number of pebbles on the graph at any time), or combinations thereof. For example cumulative space (the sum of the number of pebbles on the graph over all rounds) [5] and sustained space (the number of rounds at which many pebbles were on the graph) [6] have been suggested to model memory-hard functions. Another important distinction is between sequential and parallel strategies; a parallel player can – in every round – put as many pebbles on the graph as he wants, whereas a sequential player can put only one. For reasons discussed below, in this paper we will consider **time complexity** in the **parallel black-pebbling** model.

As pebbling is a simple combinatorial game, it’s often possible to prove unconditional lower bounds on the resources required to pebble some graphs, and in some cases one can prove that these bounds imply lower bounds for problems in more interesting computational models. In particular, if the pebbling game considered is “deterministic” in the sense that the player is initially given the graph and a designated set of nodes to pebble, then one can use an elegant proof strategy (coined “ex-post facto” in the paper [15] that introduced it) to translate basically any pebbling lower bound to a corresponding lower bound in the random oracle model for computing the “labels” of the designated set of nodes, where the label of a node is the output of the random oracle on input the labels of its parents.<sup>6</sup>

Pebbling games capture most constructions of so called memory-hard functions (MHFs), which are functions that require a lot of memory to be computed. One distinguishes between *data-independent* MHFs, where the memory access pattern is independent of the functions input, and more general *data-dependent* MHFs. The pebbling game capturing data-independent MHFs is deterministic, but for data-dependent MHFs it’s randomized, for this reason almost all security proofs for data-dependent MHFs need to make additional assumptions on the adversary. An exception is the recent security proof for the data-dependent MHF called SCRYPT [7], which proves that SCRYPT has high cumulative memory complexity in the parallel random-oracle model. Despite the fact that here the underlying pebbling game is randomized, their proof does not need to make any assumptions on the adversarial behaviour.

Like data-dependent MHFs, also the pebbling game (called  $\Phi$  and defined in §6.1) underlying pebbling-based PoS [16, 30] is randomized. Prior to this work no pebbling-based PoS had an unconditional security proof in the random oracle model; one had to assume that an adversarial prover only stores a subset of the data the honest prover would store, but not arbitrary functions of this data.<sup>7</sup>

<sup>6</sup> The high-level idea of an “ex-post-facto” proof is to look at the execution of the game in the random oracle model and translate this to a pebbling strategy, where every time a label is computed, we put a pebble on the corresponding node. Now, if the resources (where a round the pebble game translates to a round of queries to the random oracle, and a pebble translates to space required to store one label) required by the labelling game are smaller than in the derived pebbling game, we can use the adversary in the labelling game to compress the random oracle. But this is impossible as a uniformly random string cannot be compressed, so we have a contradiction, and the labelling game must have used at least as many resources as the lower bound for the corresponding pebbling game dictates.

<sup>7</sup> In [16] some combinatorial conjectures were stated which – if true – would have implied that restricting

The key observation that allows us to translate pebbling-lower bounds for a “randomized” pebbling game like  $\Phi$  to lower bounds for the “labelling game”  $\Lambda_{\text{PoS}_o}$  in the random oracle model (this game is defined in §7.1) is already implicit in [7], and goes as follows: If the complexity we consider is *time* complexity in the *parallel* black-pebbling game, then the optimal pebbling strategy is oblivious to the randomness (which in our game is a random node we need to pebble). Concretely, the strategy minimizing the number of rounds required is to put in every round a pebble on every possible node (i.e., every node whose parents are pebbled). We observe that the reason “ex-post facto” proofs can’t be done for randomized pebbling games is that the adversaries’ pebbling strategy can depend on the game’s randomness, but as just outlined, for parallel time complexity we can assume the adversary is oblivious to the randomness, and this allows us to push through a pretty standard ex-post facto type proof (proof of Theorem 7 in §7.2) showing that lower bounds on the hardness of the game  $\Phi$  imply lower bounds on the game  $\Lambda_{\text{PoS}_o}$ , which captures the security of  $\text{PoS}_o$  as a PoS. Very informally, the proof is a compression argument, which uses an adversary that is “too successful” in computing the labels of nodes it is being challenged on into a compressing encoding algorithm for the random oracle  $H$ . As a random oracle is incompressible, such an encoding cannot exist, and we get a contradiction.

In Theorem 12 in §8.2 we extend this proof from the basic  $\text{PoS}_o$  to the modes  $\text{PoCS}_\phi$  and  $\text{PoR}$ . The problem we face is that now, the values this “too successful” adversary predicts are not just outputs (i.e., labels  $\ell_i$  as in  $\text{E}_{\text{PoS}_o}$ ) of the random oracle  $H$ , but now they are of the form  $\ell_i \oplus d_i$ , where  $d_i$  is chosen by the adversary itself. Thus we can’t readily use the fact that we learned them to compress  $H$ . To solve this problem, we use the fact that in  $\text{PoCS}_\phi, \text{PoR}$ , the adversary must first commit to the  $d_i$ ’s, and this commitment is then used to sample a fresh random oracle to compute the labels. We let our encoding algorithm first runs this adversary who chooses the  $d_i$ ’s and computes the commitment. From this adversary we can extract all the  $d_i$ ’s. Once these are known, the encoding proceeds basically as for the basic  $\text{PoS}_o$ .

Extending the proof to show that our efficiently updatable  $\text{PoCS}$   $\text{PoCS}_F$  is a PoS is much more challenging, and for space reasons we only provide the proof in the full version [29], now only giving a high level idea of the challenges. In  $\text{PoCS}_F$  the labels of the “too successful” adversary predicts are of the form  $\ell_i \oplus F(d_i)$ , but the adversary has not committed to the  $d_i$ ’s before computing the  $d_i$ ’s. The key idea is to replace in the security game the random invertible function  $F : \{0, 1\}^{\lambda-w} \rightarrow \{0, 1\}^\lambda$  with the composition of two randomly sampled functions  $\mathbf{g}(f(\cdot))$ , where  $f : \{0, 1\}^{\lambda-w} \rightarrow \{0, 1\}^{w/2}$ ,  $\mathbf{g} : \{0, 1\}^{w/2} \rightarrow \{0, 1\}^\lambda$ , and argue that with high probability this game will behave like the original one (in particular, the adversary is almost as successful here). In this new game, we can recover  $\ell_i$  from the labels the adversary predicts, which now are of the form  $\ell_i \oplus \mathbf{g}(f(d_i))$ , if additionally given only the short  $w/2$  bit string  $f(d_i)$ , this is good enough to get compression and again push through an ex-post-facto type proof.

## 6 The Graph Pebbling Game $\Phi$ and its Hardness

In this section we define a pebbling game  $\Phi$  and show it’s hard if instantiated with depth-robust graphs. Later we will prove that hardness of  $\Phi$  implies hardness of games capturing the PoS security of our schemes.

---

adversaries like this is without loss of generality. But these conjectures have been beautifully refuted in [24].



## 6.1 The Pebbling Game $\Phi(\mathcal{G}, V_C)$

The game is parameterized by a DAG  $\mathcal{G} = (V, E)$ , a subset  $V_C \subseteq V$  of  $|V_C| = N$  challenge nodes, and an integer  $s, 0 \leq s \leq N$ . It is played by an adversary given as a pair  $\mathbf{A}_\Phi = \{\mathbf{A}_\Phi^1, \mathbf{A}_\Phi^2\}$ .

**initialization:**  $\mathbf{A}_\Phi^1$  gets no input, and outputs the initial pebbling configuration, which is a subset  $P_0 \subseteq V$  of  $|P_0| = s$  nodes.

**execution:** A random challenge node  $c \leftarrow V_C$  is chosen.

$\mathbf{A}_\Phi^2$  gets as input  $P_0$  and the challenge  $c$ . It then proceeds in rounds, starting at round 1.

In round  $i$ ,  $\mathbf{A}_\Phi^2$  can place (arbitrary many) pebbles on the nodes of  $V$  to update the pebbling configuration from  $P_{i-1}$  to  $P'_i$  according to the following rule: a pebble can be placed on node  $v \in V$  only if all the parents of  $v$  are pebbled in  $P_{i-1}$ . It then can remove any number of pebbles to get the configuration  $P_i \subseteq P'_i$ .

► **Definition 2** (hardness of the game  $\Phi$ ). For  $s, t \in \mathbb{N}, \epsilon \in [0, 1]$ , we say  $\mathbf{A}_\Phi$  does  $(s, t, \epsilon)$ -win the pebbling game  $\Phi(\mathcal{G}, V_C)$  (as defined above) if the probability (over the choice of  $c$  and  $\mathbf{A}_\Phi$ 's random coins) that  $\mathbf{A}_\Phi^2$  puts a pebble on  $c$  in  $t - 1$  rounds or less is at most  $\epsilon$ .

We say  $\Phi(\mathcal{G}, V_C)$  is  $(s, t, \epsilon)$ -hard if no such  $\mathbf{A}_\Phi$  exists, that is, no adversary can pebble an  $\epsilon$  fraction of  $V_C$  in  $t$  rounds or less, having only  $s$  initial pebbles.

► **Remark** (greedy is best). We observe that the optimal strategy for  $\mathbf{A}_\Phi^2$  is trivial: the greedy strategy, where in every round  $\mathbf{A}_\Phi^2$  puts pebbles on all nodes possible and never removes a pebble, is at least as good as any other strategy. This greedy strategy is oblivious to the challenge  $c$ , which will be crucial in our proofs.

## 6.2 Depth Robust Graphs

► **Definition 3** (depth-robust graphs). A DAG  $\mathcal{G} = (V, E)$  on  $V = |N|$  nodes is  $(e, d)$ -depth robust if after removing any subset of  $e \cdot N$  nodes, there remains a path of length  $d \cdot N$ .

Such graphs were first considered by Erdős et al. [17], and recent work has made them more practical, cf. [4, 6] and references therein. Concretely, for any  $\epsilon > 0$ , [6] constructs a family  $\{\mathcal{G}_N^\epsilon\}_{N \in \mathbb{N}}$  of graphs of indegree  $O(\log N)$  (here the hidden constant depends on  $\epsilon$ ) which, for any  $e, d, e + d \leq 1 - \epsilon$  are  $(e, d)$ -depth robust.

Note that any graph, even the complete graph (which has indegree  $N - 1$ ) is only  $(e, d)$  depth-robust for  $e + d = 1$ . It is maybe surprising that one gets almost as good depth-robustness as the complete graph with only  $O(\log N)$  indegree (on the negative side, it's known that  $\Omega(\log N)$  indegree is necessary for this). Let us mention that the indegree of the  $\mathcal{G}$  we use to instantiate our schemes is important as the efficiency of our schemes (in particular the proof size) depends linearly on it.

## 6.3 $\Phi(\mathcal{G}, V_C)$ is Hard if $\mathcal{G}$ is Depth Robust

Let us observe that the  $\Lambda_{\text{PoS}_0}(\mathcal{G}, V_C)$  cannot be  $(s = N \cdot c_e, t, \epsilon = c_e)$ -hard for any  $c_e \in [0, 1]$  even for tiny  $t = 1$ , as one always can simply put those  $s$  initial pebbles on an  $c_e$  fraction of  $V_C$ , and this  $\epsilon = c_e$  fraction is then already pebbled in round 1. By the lemma below, using the depth-robust graphs  $\mathcal{G}_{4N}^{\epsilon'}$  the game becomes hard – i.e. we need  $t \geq N$  rounds – for just a slightly larger fraction  $\epsilon = c_e + 4\epsilon'$ .

► **Lemma 4** (hardness of  $\Phi$  with the depth-robust graphs from [6]). *For any  $N \in \mathbb{N}, \epsilon' > 0$  consider the graph  $\mathcal{G}_{4N}^{\epsilon'}$  from [6], which is  $(e, d)$ -depth robust for any  $e + d \geq 1 - \epsilon'$ . Let  $V_C \subseteq V, |V_C| = N$  be the  $N$  topologically last nodes in  $V$ . Then, for any  $c_e \in [0, 1]$  the game  $\Phi(\mathcal{G}_{4N}^{\epsilon'}, V_C)$  is  $(s, t, \epsilon)$ -hard with*

$$s = N \cdot c_e \quad , \quad t = N \quad , \quad \epsilon = c_e + 4\epsilon'$$



## 7.2 $\Phi$ Hardness Implies $\Lambda_{\text{PoSo}}$ Hardness

Before we show that lower bounds in the pebbling game  $\Phi$  translate to lower bounds on the labelling game  $\Lambda_{\text{PoSo}}$ , let us first mention the other (trivial) direction.

Any pebbling strategy  $A_\Phi = \{A_\Phi^1, A_\Phi^2\}$  can be transformed into a labelling strategy  $A_{\text{PoSo}} = \{A_{\text{PoSo}}^1, A_{\text{PoSo}}^2\}$  which has the same parallel time complexity and which uses  $w$  bits of space in its initial state  $S_0$  for pebble in the initial state  $P_0$ . The idea is to simply have  $A_{\text{PoSo}}$  mimic  $A_\Phi$ 's strategy, computing a label whenever  $A_\Phi$  places a pebble.

► **Proposition 6** ((trivial) hardness of  $\Lambda_{\text{PoSo}}$  implies hardness of  $\Phi$ ). *If an  $A_\Phi$  exists that  $(s, t, \epsilon)$ -wins the pebbling game  $\Phi(\mathcal{G} = (V, E), V_C)$ , then an  $A_{\text{PoSo}}$  exists which  $(m, t, \epsilon, q_2^H)$ -wins the  $\Lambda_{\text{PoSo}}(\mathcal{G}, V_C, w)$  labelling game for any  $w$ ,  $q_2^H = |V|$  and*

$$m = s \cdot w$$

**Proof.** By Remark 6.1 we can assume  $A_\Phi^2$  is a “greedy” adversary who never deletes pebbles, and thus puts at most  $|V|$  pebbles on  $\mathcal{G}$  during the entire game. If  $A_\Phi^1$  outputs an initial pebbling  $P_0$ , then  $A_{\text{PoSo}}^1$  will output an initial state that contains all the labels of the pebbles in  $P_0$

$$S_0 = \{\ell_v : v \in P_0\}.$$

Note that  $|S_0| = w \cdot |P_0| \leq w \cdot s$  as claimed.  $A_{\text{PoSo}}^2$  will also be greedy, i.e., store all the labels it ever computes. In step  $i$ , when  $A_\Phi^2$  puts fresh pebbles on  $P_i \setminus P_{i-1}$ ,  $A_{\text{PoSo}}^2$  makes a parallel query to  $H_*$  to compute all the new labels  $\{\ell_v : v \in P_i \setminus P_{i-1}\}$ . In the round where  $A_\Phi^2$  puts a pebble on  $c$ ,  $A_{\text{PoSo}}^2$  can compute and output  $\ell_{\text{guess}} = \ell_c$ . ◀

Proving the other direction – that pebbling lower bounds imply lower bounds on the labelling game – is more challenging.

► **Theorem 7** (hardness of  $\Phi$  implies hardness of  $\Lambda_{\text{PoSo}}$ ). *For any  $\alpha > 0$ , if the pebbling game  $\Phi(\mathcal{G}, V_C)$  is  $(s, t, \epsilon)$ -hard, then the labelling game  $\Lambda_{\text{PoSo}}(\mathcal{G}, V_C, w)$  is  $(m, t, \epsilon, 2^{-\alpha}, q_2^H)$ -hard where*

$$m \geq s \cdot (w - 2(\log N + \log q_2^H)) - \alpha$$

Before we get to proof of this theorem, let us state what security it implies for  $\text{PoSo}$  using the hardness of  $\Phi$  as stated in Lemma 4.

► **Corollary 8** (of Thm. 7 and Lem. 4). *For  $\mathcal{G}_{4N}^{\epsilon'}$ ,  $V_C$  as in Lemma 4, and any  $c_e \in [0, 1]$ ,*

$$\Lambda_{\text{PoSo}}(\mathcal{G}_{4N}^{\epsilon'}, V_C, w) \text{ is } (m, t, \epsilon, 2^{-\alpha}, q_2^H)\text{-hard}$$

$$\text{with } m = N \cdot c_e \cdot (w - 2(\log N + \log q_2^H)) - \alpha, \quad t = N, \quad \epsilon = c_e + 4\epsilon'$$

Let us observe that the hardness as stated is basically optimal. For slightly larger  $m = N \cdot c_e$  (i.e., if we ignore the additive log terms), it means an adversary dedicating  $N \cdot (1 - 4\epsilon' - \Delta) \cdot w$  (instead  $N \cdot w$ ) space after initialization will fail to answer a  $\Delta$  fraction of the challenges in parallel time  $< N$ . Note that in  $4N = |V|$  sequential time every challenge can be answered with no storage at all by recomputing the entire labelling. As always, by challenging this adversary on  $O(1/\Delta)$  queries in parallel we can amplify the probability of the adversary failing to answer fast arbitrary close to 1.

**Proof of Theorem 7.** To prove the theorem we assume an adversary  $A_{PoS_0} = (A_{PoS_0}^1, A_{PoS_0}^2)$  exists who  $(m, t, \epsilon, 2^{-\alpha}, q_2^H)$ -wins the labelling game. Let  $\mathcal{H}, \Pr[H_* \in \mathcal{H}] \geq 2^{-\alpha}$  be the subset of  $H_*$  for which  $A_{PoS_0}$  can win the labelling game like that, i.e., using an initial state of  $\leq m$  bits, in  $\leq t$  rounds, and for an  $\geq \epsilon$  fraction of challenges where  $A_{PoS_0}^2$  makes  $\leq q_2^H$  queries to  $H_*$  (cf. Definition 5).

We will consider the random experiment where for a given  $H_* \in \mathcal{H}$ , we first run  $A_{PoS_0}^1$  to get  $S_0$ , and then we run  $A_{PoS_0}^2$  on all challenges in parallel. This will define a set  $F$  of “fresh” labels, which are labels that occur during the execution *before* they have been actually computed (and thus intuitively must somehow have been stored in the initial state  $S_0$ ). We then prove two claims.

The first shows how the above execution translates into a strategy to  $(|F|, t, \epsilon)$ -win the pebbling game, as this game is  $(s, t, \epsilon)$ -hard, we have  $|F| \geq s$ . The second claim shows how to compress  $H_*$  by almost  $|F| \cdot w$  bits when given the initial state  $S_0$ . As most functions are incompressible, we get  $m = \|S_0\| \gtrsim s \cdot w$ . We now give the detailed proof.

As outlined above, consider any  $H_* \in \mathcal{H}$ , and let  $S_0 \leftarrow A_{PoS_0}^1$  be the initial state. Let  $V'_C \subseteq V_C$  be the set of challenges which  $A_{PoS_0}^2(S_0, \cdot)$  answers correctly in  $t$  rounds or less, as  $H_* \in \mathcal{H}$  we have  $|V'_C| \geq \epsilon|V_C|$ . In the proof we'll consider two algorithms

$A_{PoS_0}^{\parallel}$  runs  $A_{PoS_0}^2$  in parallel for all possible challenges  $c \in V_C$ . Concretely,  $A_{PoS_0}^{\parallel}$  invokes  $|V_C|$  instances of  $A_{PoS_0}^2(S_0, c)$ , one for every challenge  $c \in V_C$ . In each round,  $A_{PoS_0}^{\parallel}$  collects the queries made by all the instances of  $A_{PoS_0}^2$  that have not yet terminated, then makes one parallel query to  $H_*$  containing all the collected queries, and forwards the corresponding answers to the  $A_{PoS_0}^2$  instances. We let  $A_{PoS_0}^{\parallel}$  run for  $t$  rounds, and then stop.

$\mathcal{L}_G$  computes the labels  $\ell_1, \ell_2, \dots, \ell_{|V|}$  in topological order, making sequential queries to  $H_*$ . We refer to a query that correctly computes a label as in eq.(8), i.e., a query of the form

$$\ell_i = H_*(i, \vec{\ell}_{\text{parents}(i)})$$

as a **real query**. For  $i \in V$ , we say  $i$  is **fresh** if in some round  $A_{PoS_0}^{\parallel}$  uses a label  $\ell_i$  as (part of an) input to a query or the thread  $A_{PoS_0}^2(S_0, i)$  outputs  $\ell_i$  as its guess  $\ell_{\text{guess}} = \ell_i$  (note that then  $i \in V'_C$ ) before this label  $\ell_i$  was received as output of a real query. Let  $F \subseteq V$  denote the (indices of) the fresh labels. Thus,  $\{\ell_i\}_{i \in F}$  are all the labels that appear during  $A_{PoS_0}^{\parallel}$ 's execution before they have been computed, i.e., received as output on a real query.

► **Claim 9.** *There is an adversary  $A_{\Phi}$  that  $(s', t, \epsilon)$ -wins the pebbling game  $\Phi(\mathcal{G}, V_C)$  with  $s' = |F|$  initial pebbles (thus  $|F| \geq s$ ).*

**Proof of Claim.** Consider an  $A_{\Phi}^1$  which choses an initial pebbling  $P_0 = F$ . Then  $A_{\Phi}^2$  in round  $i$  puts a pebble on  $v$  if  $A_{PoS_0}^{\parallel}$  received  $\ell_v$  as output of a real query in round  $i$ . By construction this is a valid parallel black pebbling.

We claim that this  $A_{\Phi}^2$  puts a pebble on every node in  $V'_C$  in  $t$  steps or less, and thus  $(s, t, \epsilon)$ -wins  $\Phi(\mathcal{G}, V_C)$ . To see this, consider any  $c \in V'_C$ . If  $c \in F = P_0$  it's pebbled already in round 1. Otherwise, if  $c \in V'_C \setminus F$ , the label  $\ell_{\text{guess}} = \ell_c$  output by the thread  $A_{PoS_0}^2(S_0, c)$  was not fresh, and thus must have been received as output of a real query in some round  $j \leq t$ . By construction this  $A_{\Phi}^2$  will have put a pebble on  $c$  in round no later than  $j$ . ◀

Now that we have shown  $|F| \geq s$ , the next step is to lower bound  $\|S_0\|$ , the bitlength of the initial state, in terms of  $|F|$ . For this, we show how to compress the function table of  $H_*$  given

$S_0$  by almost by almost  $|F| \cdot w$  bits. Using the fact that a random oracle is incompressible (cf. Fact 1 in §3), we'll then derive a lower bound  $\|S_0\| \gtrsim |F| \cdot w$ . Let

$$[H_*] \in \{0, 1\}^{(2^{\ell+1}-1) \times w}$$

denote the function table of  $H_* : \{0, 1\}^{\leq \ell} \rightarrow \{0, 1\}^w$ .

► **Claim 10.** *There exists an encoding (enc, dec) which correctly decodes an  $2^{-\alpha}$  fraction of the tables*

$$\Pr_{H_*}[\text{dec}(\text{enc}([H_*])) = [H_*]] \geq 2^{-\alpha}$$

and the length of the encoding is  $\|\text{enc}([H_*], S_0)\| \leq \|[H_*]\| + \|S_0\| - |F| \cdot (w - 2(\log N + \log q_2^H))$ .

Before we prove this claim, let us observe this implies the statement of the theorem by using Fact 1, which implies

$$\|S_0\| \geq |F| \cdot (w - 2(\log N + \log q_2^H)) - \alpha.$$

**Proof of Claim.** The encoding enc/dec will correctly decode all the  $[H_*]$  which are in  $\mathcal{H}$ . For this,  $\text{enc}([H_*])$  first determines if  $H_* \in \mathcal{H}$ , and if this is not the case outputs whatever (say the bit 0). Let  $B$  denote the following computation: we first invoke  $A_{\text{PoSo}}^{\parallel}(S_0, V_C)$  followed by  $L_G$ , we'll denote with  $q \leq N \cdot (q_2^H + 1)$  the number of distinct  $H_*$  queries made during  $B$ 's execution (at most  $q_2^H$  for each invocation of the  $N$  invocations of  $A_{\text{PoSo}}^{\parallel}$  and  $N$  more for  $L_G$ ).

Let the list  $\vec{c}$  contain all the outputs of  $H_*$  queries made during  $B$ . The outputs in  $\vec{c}$  are stored in the order the queries were made, and if a query is repeated, the output is not stored. Let  $\bar{\vec{c}}$  denote the function table of  $H_*$  with the  $|\vec{c}|$   $w$ -bit entries that are in  $\vec{c}$  removed. Note that given  $\vec{c}, \bar{\vec{c}}, S_0, V_{C'}$  we can recover  $[H_*]$  by running  $B$  using  $\vec{c}$  to answer all the oracle queries. After this, we have learned all the inputs corresponding to the outputs stored in  $\vec{c}$ , and thus know which queries were deleted from  $[H_*]$  to get  $\bar{\vec{c}}$ . Thus now we can recover all of  $[H_*]$ . We haven't compressed anything yet (as  $\|\vec{c}\| + \|\bar{\vec{c}}\| = \|[H_*]\|$ , or as all elements in those sets and the table are  $w$  bit strings, equivalently  $|\vec{c}| + |\bar{\vec{c}}| = \|[H_*]\|$ ). Next we'll show how to compress  $\vec{c}$  into a smaller  $\vec{c}_F$  which, with some short extra information  $\vec{b}_F$ , will suffice to answer all  $H_*$  queries made during  $B$  correctly.

Recall that  $F \subseteq V$  are the fresh queries. Consider  $i \in F$ , at some point during the evaluation of  $B$  the real query  $\ell_i = H_*(i, \vec{\ell}_{\text{parents}(i)})$  is made (the only reason we invoke  $L_G$  as part of  $B$  is to ensure this query is made at some point). As  $i \in F$ , at the point where this query is made for the first time, we have already observed the value  $\ell_i$  as part of some query input. Let  $\vec{c}_F$  denote  $\vec{c}$ , but with the  $|F|$  entries corresponding to the real queries of  $i \in F$  deleted. With this  $\vec{c}_F$  we can answer all of  $B$ 's queries if we're given some extra information which, for every  $i \in F$ , tells as at which point during the execution of  $B$  we observe  $\ell_i$ , and where the corresponding real query is made. This extra information requires at most  $2 \log q$  bits for every  $i \in F$ , let  $\vec{b}_F$  denote a string encoding this information, we now define the encoding as

$$\text{enc}([H_*]) = (S_0, \vec{c}_F, \vec{b}_F, \bar{\vec{c}})$$

The decoding  $\text{dec}(S_0, \vec{c}_F, \vec{b}_F, \bar{\vec{c}})$  reconstructs  $[H_*]$  as outlined above. As

$$\begin{aligned} \|\vec{c}_F\| + \|\bar{\vec{c}}\| &= \|[H_*]\| - w \cdot |F| \\ \|\vec{b}_F\| &\leq |F| \cdot 2 \log q \leq |F| \cdot 2(\log N + \log q) \end{aligned}$$

the encoding length is  $\|\text{enc}([H_*], S_0)\| \leq \|S_0\| + \|[H_*]\| - |F| \cdot (w - 2(\log N + \log q))$  as claimed. ◀



## 8 PoS Security of PoCS $_\phi$ and PoR

In this section we extend the result from the previous section, and show that hardness of  $\Phi$  implies hardness of games  $\Lambda_{\text{PoCS}_\phi}$  and  $\Lambda_{\text{PoR}}$ , which capture the PoS security of our constructions PoCS $_\phi$  and PoR. We start with defining the games

### 8.1 The Labelling Games $\Lambda_{\text{PoCS}_\phi}$ and $\Lambda_{\text{PoR}}$

Let  $\text{PoXX} \in \{\text{PoCS}_\phi, \text{PoR}\}$ . The game is parameterized by a DAG  $\mathcal{G} = (V, E)$ , a subset  $V_C \subseteq V$  of  $|V_C| = N$  challenge nodes and a block size  $w$ . Moreover a function  $H_* : \{0, 1\}^{\leq \iota} \rightarrow \{0, 1\}^w$ ,  $\iota = (\delta + 2) \cdot w$ . The game is played by an adversary  $A_{\text{PoXX}} = \{A_{\text{PoXX}}^1, A_{\text{PoXX}}^2\}$ .

**initialization:**  $A_{\text{PoXX}}^1$  is given oracle access to  $H_*$ . It can choose any data  $\vec{d} = \{d_i\}_{i \in V_C}$ ,  $d_i \in \{0, 1\}^w$ , which defines labels  $\vec{\ell}$  to store as in eq.(3) and eq.(4), but using  $H_*$  instead  $H_X$ . Recall that for this we first compute  $(\phi_{\vec{d}}, \phi_{\vec{d}}^+) := \text{commit}^{H_*}(\vec{d})$ , now let  $H_{*, \phi_{\vec{d}}}$  be the function  $H_{*, \phi_{\vec{d}}}(\cdot) \equiv H_*(\phi_{\vec{d}}, \cdot)$ , and then compute  $\vec{\ell}$  as

$$\text{(if PoXX = PoR)} \quad \vec{\ell} = \{\ell_i\}_{i \in V_C} \text{ where } \ell_i = \begin{cases} H_{*, \phi_{\vec{d}}}(i, \vec{\ell}_{\text{parents}(i)}) & \text{if } i \in V \setminus V_C \\ H_{*, \phi_{\vec{d}}}(i, \vec{\ell}_{\text{parents}(i)}) \oplus d_i & \text{if } i \in V_C \end{cases}$$

$$\text{(if PoXX = PoCS}_\phi) \quad \vec{\ell} = \{\ell'_i\}_{i \in V_C} \text{ where } \forall i \in V : \ell_i = H_{*, \phi_{\vec{d}}}(i, \vec{\ell}_{\text{parents}(i)}) \\ \forall i \in V_C : \ell'_i = \ell_i \oplus d_i$$

$A_{\text{PoXX}}^1$  outputs a string (the initial state)  $S_0$  of length  $\|S_0\| = m$  bits.

**execution:** A random challenge node  $c \leftarrow V_C$  is chosen.

$A_{\text{PoXX}}^2$  gets as input  $S_0$  and challenge  $c$ . It then proceeds in rounds starting with  $i = 1$ : In round  $i$ ,  $A_{\text{PoXX}}$  gets as input its state  $S_{i-1}$ . It can either decide to stop the game by outputting a single guess  $\ell_{\text{guess}}$  (for  $\ell_c$  in PoR or  $\ell'_c$  in PoCS $_\phi$ ), or it can make one parallel oracle query: on query  $(x_1, \dots, x_{q_i})$  it receives  $(y_1, \dots, y_{q_i})$  where  $y_i = H_*(x_i)$ . It can do any amount of computation before and after this query, and at the end of the round output its state  $S_i$  for the next round.

► **Definition 11** (hardness of the games  $\Lambda_{\text{PoXX}} \in \{\Lambda_{\text{PoR}}, \Lambda_{\text{PoCS}_\phi}\}$ ). For  $m, t, q_1^H, q_2^H, w \in \mathbb{N}$  and  $p_H, \epsilon \in [0, 1]$ , we say  $A_{\text{PoXX}} = (A_{\text{PoXX}}^1, A_{\text{PoXX}}^2)$  does  $(m, t, \epsilon, p_H, q_1^H, q_2^H)$ -win the labelling game  $\Lambda_{\text{PoXX}}(\mathcal{G}, V_C, w)$  (as defined above) if for all but a  $p_H$  fraction of  $H_*$  the following holds: for at most an  $\epsilon$  fraction of challenges  $c$ ,  $A_{\text{PoXX}}^2$  correctly guesses  $c$ 's label (i.e.,  $\ell_{\text{guess}} = \ell_c$ ) in round  $t$  or earlier. Moreover  $A_{\text{PoXX}}^1$  and  $A_{\text{PoXX}}^2$  make at most  $q_1^H$  and  $q_2^H$  queries to  $H_*$ , respectively. We say  $\Lambda_{\text{PoXX}}(\mathcal{G}, V_C, w)$  is  $(m, t, \epsilon, p_H, q_1^H, q_2^H)$ -hard if no such  $A_{\text{PoXX}}$  exists.

### 8.2 $\Phi$ Hardness Implies $\Lambda_{\text{PoR}}$ and $\Lambda_{\text{PoCS}_\phi}$ Hardness

► **Theorem 12** (hardness of  $\Phi$  implies hardness of  $\Lambda_{\text{PoR}} \& \Lambda_{\text{PoCS}_\phi}$ ). For any  $\alpha > 0$ , if the pebbling game  $\Phi(\mathcal{G}, V_C)$  is  $(s, t, \epsilon)$ -hard, then the labelling game  $\Lambda_{\text{PoR}}(\mathcal{G}, V_C, w)$  and also the labelling game  $\Lambda_{\text{PoCS}_\phi}(\mathcal{G}, V_C, w)$  is  $(m, t, \epsilon, 2^{-\alpha} + q_1^H/2^w, q_1^H, q_2^H)$ -hard where

$$m \geq s \cdot (w - 2(\log N + \log q_2^H)) - \alpha$$

Before we get to proof of this theorem, let us state what security it implies for PoR and PoCS $_\phi$  using the hardness of  $\Phi$  as stated in Lemma 4.

► **Corollary 13** (of Thm. 7 and Lem. 4). For  $\mathcal{G}_{4N}^{\epsilon'}$ ,  $V_C$  as in Lemma 4, and any  $c_e \in [0, 1]$ ,

$$\Lambda_{\text{PoCS}_\phi}(\mathcal{G}_{4N}^{\epsilon'}, V_C, w) \text{ and } \Lambda_{\text{PoR}}(\mathcal{G}_{4N}^{\epsilon'}, V_C, w) \text{ are } (m, t, \epsilon, 2^{-\alpha}, q_2^H)\text{-hard}$$

with  $m = N \cdot c_e \cdot (w - 2(\log N + \log q_2^H)) - \alpha$ ,  $t = N$ ,  $\epsilon = c_e + 4\epsilon'$

**Proof.** We assume the reader is familiar with the proof of Theorem 7, as we will only explain how that proof needs to be adapted.

The proof of Theorem 7 goes through almost unchanged for  $\text{PoXX} \in \{\text{PoR}, \text{PoCS}_\phi\}$  instead of  $\text{PoS}_o$ , the point where it fails is when we need to compress fresh labels. In the proof of Theorem 7 every fresh label  $\ell_i, i \in F$  allowed us to compress one element of  $H_*$ . Now the situation is seemingly more complicated. For concreteness, let's consider  $\text{PoR}$ . Now even if the encoding  $\text{enc}$  observes a fresh label  $\ell_i$  when invoking  $A_{\text{PoR}}^{\parallel}$  (which is defined analogous to  $A_{\text{PoS}_o}^{\parallel}$  in the proof of Theorem 7), it's not clear how to compress one entry of  $H_{*, \phi_{\vec{d}}}$ 's function table as now

$$\ell_i = H_{*, \phi_{\vec{d}}}(i, \vec{\ell}_{\text{parents}(i)}) \oplus d_i$$

only provides an output that is blinded with  $d_i$ . If we could make sure the encoding and decoding  $\text{enc}/\text{dec}$  knew the  $\vec{d}$ , this problem would disappear. We fix this problem as follows. We define  $A_{\text{PoR}}^{\parallel}$  analogous to  $A_{\text{PoS}_o}^{\parallel}$ , i.e., it runs  $A_{\text{PoR}}^2(S_0, c)$  on all challenges  $c \in V_C$  in parallel. But additionally, at the very beginning (before invoking the  $A_{\text{PoR}}^2$ 's), it invokes  $A_{\text{PoR}}^1$ , but only runs it to the point where the commitment  $\phi_{\vec{d}}$  is received as an output of  $H_*$  (recall we assume  $A_{\text{PoR}}^1$  follows the protocol, so this commitment must be computed at some point). This way  $\text{enc}/\text{dec}$ , we invoke  $A_{\text{PoR}}^{\parallel}$ , learn the entire  $\vec{d}$ , as it can be extracted from the  $H_*$  queries leading to  $\phi_{\vec{d}}$ . At the same time  $A_{\text{PoR}}^1$  will almost certainly not have made any  $H_{*, \phi_{\vec{d}}}(\cdot) = H_*(\phi_{\vec{d}}, \cdot)$  queries as  $\phi_{\vec{d}}$  is uniform and we stop executing  $A_{\text{PoR}}^1$  once  $\phi_{\vec{d}}$  is received. This is necessary, so a label that was fresh without running  $A_{\text{PoR}}^1$  first, will still be fresh if we do run  $A_{\text{PoR}}^1$ . The above argument works as long as  $A_{\text{PoR}}^1$  doesn't find a collision in  $H_*$  (otherwise we can't extract a unique  $\vec{d}$ ). For this reason in the theorem security holds only for a slightly smaller  $p_H = 2^{-\alpha} + q_1^{H^2}/2^w$  fraction of the  $H_*$  than the  $p_H = 2^{-\alpha}$  fraction we got for the  $\Lambda_{\text{PoS}_o}$  game in Theorem 7. ◀

---

## References

- 1 Chia Network. <https://chia.network/>, 2017.
- 2 Burstcoin. URL: <http://burstcoin.info>.
- 3 Hamza Abusalah, Joël Alwen, Bram Cohen, Danylo Khilko, Krzysztof Pietrzak, and Leonid Reyzin. Beyond Hellman's Time-Memory Trade-Offs with Applications to Proofs of Space. In Tsuyoshi Takagi and Thomas Peyrin, editors, *ASIACRYPT 2017, Part II*, volume 10625 of *LNCS*, pages 357–379. Springer, Heidelberg, December 2017.
- 4 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 17*, pages 1001–1017. ACM Press, October / November 2017.
- 5 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.
- 6 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.
- 7 Joël Alwen, Binyi Chen, Krzysztof Pietrzak, Leonid Reyzin, and Stefano Tessaro. Script Is Maximally Memory-Hard. In Jean-Sébastien Coron and Jesper Buus Nielsen, editors, *EUROCRYPT 2017, Part III*, volume 10212 of *LNCS*, pages 33–62. Springer, Heidelberg, April / May 2017.
  - 8 Marshall Ball, Alon Rosen, Manuel Sabin, and Prashant Nalini Vasudevan. Proofs of Useful Work. *IACR Cryptology ePrint Archive*, 2017:203, 2017.
  - 9 Harry Buhrman, Richard Cleve, Michal Koucký, Bruno Loff, and Florian Speelman. Computing with a full memory: catalytic space. In David B. Shmoys, editor, *46th ACM STOC*, pages 857–866. ACM Press, May / June 2014.
  - 10 Harry Buhrman, Michal Koucký, Bruno Loff, and Florian Speelman. Catalytic Space: Non-determinism and Hierarchy. In *STACS*, volume 47 of *LIPICs*, pages 24:1–24:13. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2016.
  - 11 Miles Carlsten, Harry A. Kalodner, S. Matthew Weinberg, and Arvind Narayanan. On the Instability of Bitcoin Without the Block Reward. In Edgar R. Weippl, Stefan Katzenbeisser, Christopher Kruegel, Andrew C. Myers, and Shai Halevi, editors, *ACM CCS 16*, pages 154–167. ACM Press, October 2016.
  - 12 Phil Daian, Rafael Pass, and Elaine Shi. Snow White: Provably Secure Proofs of Stake. *Cryptology ePrint Archive*, Report 2016/919, 2016. URL: <http://eprint.iacr.org/2016/919>.
  - 13 Anindya De, Luca Trevisan, and Madhur Tulsiani. Time Space Tradeoffs for Attacks against One-Way Functions and PRGs. In Tal Rabin, editor, *CRYPTO 2010*, volume 6223 of *LNCS*, pages 649–665. Springer, Heidelberg, August 2010.
  - 14 Yevgeniy Dodis, Siyao Guo, and Jonathan Katz. Fixing Cracks in the Concrete: Random Oracles with Auxiliary Input, Revisited. In Jean-Sébastien Coron and Jesper Buus Nielsen, editors, *EUROCRYPT 2017, Part II*, volume 10211 of *LNCS*, pages 473–495. Springer, Heidelberg, April / May 2017.
  - 15 Cynthia Dwork, Moni Naor, and Hoeteck Wee. Pebbling and Proofs of Work. In Victor Shoup, editor, *CRYPTO 2005*, volume 3621 of *LNCS*, pages 37–54. Springer, Heidelberg, August 2005.
  - 16 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.
  - 17 Paul Erdős, Ronald L. Graham, and Endre Szemerédi. On Sparse Graphs with Dense Long Paths. Technical report, Stanford University, Stanford, CA, USA, 1975.
  - 18 Ben Fisch. PoReps: Proofs of Space on Useful Data. *Cryptology ePrint Archive*, Report 2018/678, 2018. URL: <https://eprint.iacr.org/2018/678>.
  - 19 Ben Fisch. Tight Proofs of Space and Replication. *Cryptology ePrint Archive*, Report 2018/702, 2018. URL: <https://eprint.iacr.org/2018/702>.
  - 20 Aggelos Kiayias, Alexander Russell, Bernardo David, and Roman Oliynykov. Ouroboros: A Provably Secure Proof-of-Stake Blockchain Protocol. In Jonathan Katz and Hovav Shacham, editors, *CRYPTO 2017, Part I*, volume 10401 of *LNCS*, pages 357–388. Springer, Heidelberg, August 2017.
  - 21 Eike Kiltz, Krzysztof Pietrzak, and Mario Szegedy. Digital Signatures with Minimal Overhead from Indifferentiable Random Invertible Functions. In Ran Canetti and Juan A. Garay, editors, *CRYPTO 2013, Part I*, volume 8042 of *LNCS*, pages 571–588. Springer, Heidelberg, August 2013. doi:10.1007/978-3-642-40041-4\_31.
  - 22 Sunny King and Scott Nadal. Ppcoin: Peer-to-peer crypto-currency with proof-of-stake.

- 23 Protocol Labs. Filecoin: A Decentralized Storage Network. <https://filecoin.io/filecoin.pdf>, 2017.
- 24 Daniel Malinowski and Karol Zebrowski. Disproving the Conjectures from “On the Complexity of Scrypt and Proofs of Space in the Parallel Random Oracle Model”. In *ICITS*, volume 10681 of *Lecture Notes in Computer Science*, pages 26–38. Springer, 2017.
- 25 Silvio Micali. ALGORAND: the efficient and democratic ledger. *CoRR*, abs/1607.01341, 2016.
- 26 Andrew Miller, Ari Juels, Elaine Shi, Bryan Parno, and Jonathan Katz. Permacoin: Repurposing Bitcoin Work for Data Preservation. In *2014 IEEE Symposium on Security and Privacy*, pages 475–490. IEEE Computer Society Press, May 2014. doi:10.1109/SP.2014.37.
- 27 Sunoo Park, Krzysztof Pietrzak, Albert Kwon, Joël Alwen, Georg Fuchsbauer, and Peter Gaži. SpaceMint: A Cryptocurrency Based on Proofs of Space. Cryptology ePrint Archive, Report 2015/528, 2015. URL: <https://eprint.iacr.org/2015/528>.
- 28 Wolfgang J. Paul, Robert Endre Tarjan, and James R. Celoni. Space bounds for a game on graphs. *Mathematical systems theory*, 10(1):239–251, 1976–1977.
- 29 Krzysztof Pietrzak. Proofs of Catalytic Space. Cryptology ePrint Archive, Report 2018/194, 2018. , full version of this paper. URL: <https://eprint.iacr.org/2018/194>.
- 30 Ling Ren and Srinivas Devadas. Proof of Space from Stacked Expanders. In Martin Hirt and Adam D. Smith, editors, *TCC 2016-B, Part I*, volume 9985 of *LNCS*, pages 262–285. Springer, Heidelberg, October /November 2016. doi:10.1007/978-3-662-53641-4\_11.
- 31 The NXT Community. Nxt Whitepaper. <https://bravenewcoin.com/assets/Whitepapers/NxtWhitepaper-v122-rev4.pdf>, July 2014.

## **A** Discussion and Motivation

### **A.1** The Quest for a Sustainable Blockchain

PoW based blockchains, most notably Bitcoin, have been criticized as the mining process (required to secure the blockchain) results in a massive energy waste. This is not only problematic ecologically, but also economically, as it requires high rewards for the miners to compensate for this energy loss.<sup>8</sup>

#### **A.1.1** Proofs of Stake

The idea behind “Nakamoto consensus” used in Bitcoin, is to randomly chose a miner to generate the next block, where the probability of any miner to be chosen is proportional to its hashing power. The most investigated idea to replace PoWs in blockchains are “proofs of stake” (PoStake), where the idea is to choose the winner proportional to the fraction of coins they hold. At first, this idea looks promising, but it seems to be difficult to actually realize it in a secure and efficient way. Early ad-hoc implementations of this idea include Peercoin [22] and NXT [31]. More recent proposals come with security proofs in various models [25, 12, 20], but those protocols are fairly complicated, and basically run a byzantine agreement protocol amongst rotating subsets of the miners, thus losing the appealing simplicity of Bitcoin, where a winning miner simply gossips the next block and no other interaction is required.

---

<sup>8</sup> The block reward currently used as main compensation for miners in Bitcoin is decreasing (it’s halved every four years), and thus ultimately will be replaced with only transaction fees, which might create serious problems [11].

### A.1.2 Space as a Resource

After time, space is the best investigated resource in computational complexity, it's thus only natural to try using disk space as a resource for mining. This has the potential to give blockchains which are much more sustainable than PoW based designs, while avoiding at least some of the technical issues that PoStake has. Permacoin [26] requires a miner to dedicate disk space, but it's still a proof of work based design, only now the computation itself (which is a so called "proof of retrievability") requires access to a large disk. A proposal which mostly uses space as resource is Burstcoin [2], this design is poorly document, but it seems to have security and efficiency issues.<sup>9</sup> Another suggestion are "proofs of space" (PoS), which are the topic of this paper and we'll discuss them in more detail below in §A.1.4.

### A.1.3 Useful Proofs

While PoStake aim to avoid wasting significant resources for mining in the first place, another approach to minimize the footprint of mining is to use the resources required to sustain the blockchain for something useful. A intriguing idea is to use the computing power wasted for PoWs for solving actual computational problems, we refer the reader to [8] and the references therein. In this work we also follow this approach and construct "proofs of catalytic space" (PoCS), which are defined like PoS, but where most of the space required by the prover can be used to store useful data.

### A.1.4 Proofs of Space (PoS)

Proofs of space (PoS) [16] are proofs systems that were developed to serve as a replacement for PoW in blockchain designs. The first proposal of a PoS-based blockchain is Spacemint [27], a recent ongoing effort which combines PoS with some type of proofs of sequential work is the Chia network [1]. A PoS [16] is a two stage protocol between a prover  $P$  and a verifier  $V$ . The first phase is an initialization protocol which is run only once, after which  $P$  has initialized its space. Then there's a proof execution phase which typically is run many times, in which an honest prover  $P$  can *efficiently* convince the verifier that dedicates the space. The verifier  $V$  is required to be very efficient during both phases, this means it can be polynomial in some security parameter, but should be almost independent (i.e., depend at most polylogarithmically) on the size  $N$  of the space committed by the prover. The honest prover  $P$  is required to be very efficient during the execution phases. During the initialization  $P$  cannot be very efficient, as it must at the very least overwrite all of the claimed space, but it shouldn't require much more than that.

To date two very different types of PoS have been suggested. PoS-based on hard to pebble graphs [16, 30] and PoS-based on inverting random functions [3], the new proofs systems – for proofs of catalytic space (PoCS) and proofs of replication (PoR) – we propose in this work extend the pebbling-based PoS. We leave it as an open problem to extend

---

<sup>9</sup> [27, Appendix B of the full version] discusses some issues with (our best guess on what is) Burst and the underlying proof system called "proofs of capacity" (PoC). In a nutshell, PoC are rather inefficient as the prover needs to access a constant (albeit small) fraction of the entire space for generating a proof, and verification requires over a Million hashes. As to security, PoC allow for strong time-memory trade-offs (a recent ad-hoc fix <https://www.burst-coin.org/wp-content/uploads/2017/07/The-Burst-Dymaxion-1.00.pdf> claims to address at least the most obvious time-memory attacks outlined in [27]). But most worryingly, the blockchain designs seems to have no mechanisms to address nothing-at-stake issues, which are responsible for the most delicate and complicated issue of any blockchain design not based on proofs of work.

the [3] PoS to PoR and PoCS, this would be very interesting as although the [3] PoS has worse *asymptotic* security than pebbling-based PoS, but it's much more efficient (with proofs of length a few hundred bits, and proof generation and verification requiring just a small constant number of hash queries). Moreover, unlike pebbling-based PoS, this PoS has a non-interactive initialization phase, which makes it easier to use it for a blockchain design where we have no dedicated verifier, and thus the proof must be made non-interactive.<sup>10</sup>

---

<sup>10</sup> Concretely, for subtle security reasons Spacemint [27] (which uses the pebbling-based PoS) requires the miners to commit to the transcript of a challenge response protocol which is run during the initialization phase. This is done by uploading this (short) transcript to the blockchain. The Chia network which are based on [3] will not require any such commitments.



# Simple Verifiable Delay Functions

Krzysztof Pietrzak<sup>1</sup>

Institute of Science and Technology Austria, Austria  
pietrzak@ist.ac.at

---

## Abstract

---

We construct a verifiable delay function (VDF) by showing how the Rivest-Shamir-Wagner time-lock puzzle can be made publicly verifiable.

Concretely, we give a statistically sound public-coin protocol to prove that a tuple  $(N, x, T, y)$  satisfies  $y = x^{2^T} \pmod{N}$  where the prover doesn't know the factorization of  $N$  and its running time is dominated by solving the puzzle, that is, compute  $x^{2^T}$ , which is conjectured to require  $T$  sequential squarings. To get a VDF we make this protocol non-interactive using the Fiat-Shamir heuristic.

The motivation for this work comes from the Chia blockchain design, which uses a VDF as a key ingredient. For typical parameters ( $T \leq 2^{40}$ ,  $N = 2048$ ), our proofs are of size around  $10KB$ , verification cost around three RSA exponentiations and computing the proof is 8000 times faster than solving the puzzle even without any parallelism.

**2012 ACM Subject Classification** Theory of computation  $\rightarrow$  Cryptographic primitives, Theory of computation  $\rightarrow$  Interactive proof systems

**Keywords and phrases** Verifiable delay functions, Time-lock puzzles

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2019.60

**Related Version** <https://eprint.iacr.org/2018/627.pdf>

## 1 Introduction

### 1.1 The RSW time-lock puzzle

Rivest, Shamir and Wagner [10] introduced the concept of a time-lock puzzle, and proposed the following elegant construction

**The puzzle** is a tuple  $(N, x, T)$  where  $N = p \cdot q$  is an RSA modulus,  $x \in \mathbb{Z}_N^*$  is random and  $T \in \mathbb{N}$  is a time parameter.

**The solution** of the puzzle is  $y = x^{2^T} \pmod{N}$ . It can be computed making two exponentiations by the party who generates the puzzle (and thus knows the group order  $\phi(N) = (p-1)(q-1)$ ) as

$$e := 2^T \pmod{\phi(N)} \quad , \quad y := x^e \pmod{N} \tag{1}$$

but is conjectured to require  $T$  sequential squarings if the group order (or equivalently, the factorization of  $N$ ) is not known

$$x \rightarrow x^2 \rightarrow x^{2^2} \rightarrow x^{2^3} \rightarrow \dots \rightarrow x^{2^T} \pmod{N} \tag{2}$$

---

<sup>1</sup> 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 682815/TOCNeT).



To be more precise, the conjecture here is that  $T$  sequential steps are necessary to compute  $x^{2^T} \pmod N$  even if one can use large parallelism.

As an application, [10] show how to “encrypt to the future”: sample a puzzle  $(N, x, T)$  together with its solution  $y$ , then derive a key  $k_y$  from  $y$  and encrypt a message  $m$  into a ciphertext  $c = \text{ENC}(k_y, m)$ . Given  $(N, x, T)$  and  $c$  one can recover the message  $m$  in time required to compute  $T$  squarings sequentially, but (under the above conjecture) not faster.

## 1.2 Proofs of sequential work (PoSW)

Proofs of sequential work (PoSW) are closely related to time-lock puzzles. PoSW were introduced in [9], and informally are proof systems where on input a random challenge  $x$  and time parameter  $T$  one can compute a *publicly verifiable* proof making  $T$  sequential computations, but it’s hard to come up with an accepting proof in significantly less than  $T$  sequential steps, even given access to massive parallelism.

The PoSW constructed in [9] is not very practical (at least for large  $T$ ) as the prover needs not only  $T$  time, but also linear in  $T$  space to compute a proof. Recently [5] constructed a very simple and practical PoSW in the random oracle model. They were interested in PoSW as they serve as a key ingredient in the Chia blockchain design ([chia.net](https://chia.net)).

The main open problem left open in [5] was to construct PoSW that is *unique*, in the sense that one cannot compute two accepting proofs on the same challenge. The existing PoSW all allow to generate many accepting proofs at basically the same cost as honestly computing the proof. Unfortunately such PoSW cannot be used for the blockchain application just mentioned, as this would allow for so called grinding attacks. More precisely, the output of the PoSW is used to compute a challenge for generating the next block. If the PoSW is not unique, a malicious miner could compute many proofs, and then pick the one which results in a challenge that is most favourable for him.

## 1.3 Verifiable delay functions (VDF)

Boneh, Bonneau, Bünz and Fisch [3] recently introduced the notion of a verifiable delay function (VDF). A VDF can be seen as a relaxation of unique PoSW which still suffices for all known applications of unique PoSW. We refer the reader to [3] for a thorough discussion on VDFs including many interesting applications. In a VDF the proof on challenge  $(x, T)$  has two parts  $(y, \pi)$ , where  $y$  is a deterministic function of  $x$  that needs  $T$  sequential time to compute, and  $\pi$  is a proof that  $y$  was correctly computed (the reason this is not necessarily a unique PoSW is the fact that this  $\pi$  does not need to be unique). It must be possible to compute  $\pi$  with low parallelism and such that  $\pi$  can be output almost at the same time as  $y$ . In [3] this is achieved using incrementally verifiable computation [12]. The (very high level) idea is to compute a hash chain

$$y = \underbrace{h(h(\dots h(x) \dots))}_{T \text{ times}}$$

and at the same time use incrementally verifiable computation to compute the proof  $\pi$ , so the proof will be ready shortly after  $y$  is computed. To make this generic approach actually practical the  $h$  used in [3] is a particular algebraic function (a permutation polynomial) which has the property that one can invert it significantly faster than compute in forward direction (so instead of verifying the evaluation of  $h(\cdot)$ , one can just verify the much simpler computation of  $h^{-1}(\cdot)$ ), and also the proof system used to compute  $\pi$  is tailored so it can exploit the algebraic structure of  $h$ .



## 1.4 A VDF from RSW

The RSW time-lock puzzle looks like a promising starting point for constructing a VDF. The main difficulty one needs to solve is achieving public verifiability: to efficiently verify  $y \stackrel{?}{=} x^{2^T} \pmod{N}$  one needs the group order of  $\mathbb{Z}_N^*$  (or equivalently, the factorization of  $N$ ). But the factorization cannot be public as otherwise also computing  $y$  becomes easy.

One idea to solve this issue is to somehow obfuscate the group order so it can only be used to efficiently verify if a given solution is correct, but not to speed up its computation. There currently is no known instantiation to this approach.

In this work we give a different solution. We construct a protocol where a prover  $\mathcal{P}$  can convince a verifier  $\mathcal{V}$  it computed the correct solution  $y = x^{2^T} \pmod{N}$  without either party knowing the factorization (or any other hard to compute function) of  $N$ . Our interactive protocol is public-coin, but can be made non-interactive – and thus give a VDF – by the Fiat-Shamir transformation. Here the prover’s messages are replaced by simply applying a random function to the transcript. The Fiat-Shamir transformation applied to any constant-round public-coin interactive proof systems results in a sound non-interactive proof system in the random oracle model. Although our proof is not constant-round, we can still show that this transformation works, i.e., gives a sound non-interactive proof system relative to a random function (i.e., in the random oracle model). In practice the random function is instantiated with an actual hash-function like SHA256, as then soundness only holds computationally, such systems are called arguments, not proofs.

Our protocol is inspired by the sumcheck protocol [8, 11]. The key idea of the proof is very simple. Assume  $\mathcal{P}$  wants to convince  $\mathcal{V}$  that a tuple  $(x, y)$  satisfies  $y = x^{2^T}$ . For this,  $\mathcal{P}$  first sends  $\mu = x^{2^{T/2}}$  to  $\mathcal{V}$ . Now  $\mu = x^{2^{T/2}}$  together with  $y = \mu^{2^{T/2}}$  imply  $y = x^{2^T}$ . The only thing we have achieved at this point is to reduce the time parameter from  $T$  to  $T/2$  at the cost of having two instead just one statement to verify. We then show that the verifier can merge those two statements in a randomized way into a single statement  $(x', y') = (x^r \cdot \mu, \mu^r \cdot y)$  that satisfies  $y' = x'^{2^{T/2}}$  if the original statement  $y = x^{2^T}$  was true (and  $\mathcal{P}$  sends the correct  $\mu$ ), but is almost certainly wrong (over the choice of the random exponent  $r$ ) if the original statement was wrong, no matter what  $\mu$  the malicious prover did send. This subprotocol is repeated  $\log(T)$  times – each time halving the time parameter  $T$  – until  $T = 1$ , at which point  $\mathcal{V}$  can efficiently verify correctness of the claim itself.

The VDF we get has short proofs and is efficiently verifiable. For typical parameters (2048 bit modulus and  $\log(T) \leq 40$ ) a proof is about 10KB large and the cost for verification is around three full exponentiations (for comparison, a standard RSA decryption or RSA signature computation requires one full exponentiation).

The algebraic setting of our proof systems differs a bit from RSW, as we’ll discuss in §2. In a nutshell, to prove statistical soundness, we need to assume that  $N$  is the product of two *safe* primes, i.e.,  $N = p \cdot q$  where  $p' = (p - 1)/2$  and  $q' = (q - 1)/2$  are prime, as then a random quadratic residue  $x \in QR_N$  almost certainly is a generator of  $QR_N$ . We’ll actually preform all computations in the group of *signed* quadratic residues  $QR_N^+$ , as unlike for  $QR_N$ , one can efficiently decide if an element is in  $QR_N^+$ , which will make the protocol slightly simpler and more efficient. Using  $QR_N^+$  instead of  $QR_N$  will also make the proof unique, so our VDF is a unique PoSW.

## 1.5 Wesolowski's VDF

A closely related result to our VDF is a concurrent paper by Wesolowski [14]. A recent survey [4] compares his construction with the one presented in this paper.

Wesolowski also constructs a VDF by making the RSW time-lock puzzle publicly verifiable. The prover, who claims  $y = x^{2^T}$ , receives as challenge a large prime  $B$ , and must respond with the proof  $\pi = x^{\lfloor \frac{2^T}{B} \rfloor}$ . To verify this proof one checks  $\pi^B \cdot x^{2^T \bmod B} \stackrel{?}{=} y$ .<sup>2</sup> To get a VDF one makes this protocol non-interactive using the Fiat-Shamir heuristic, i.e.,  $B = \text{hash}(y)$  is computed as a hash of the first message.

To prove soundness (i.e., that it's hard to come up with a  $z \neq x^{2^T}$  together with a  $\pi$  that passes verification) one needs a computational hardness assumption which basically states that for any  $z \neq 1$  it is hard to compute the  $B$ 'th root of  $z$  (i.e. a  $y$  s.t.  $z^B = y$ ) in the underlying group when  $B$  is a large random prime (whereas soundness of our proof is unconditional).

The main advantage of Wesolowski's construction over ours is that his proof is just a single group element, and thus about a  $\log(T)$  factor smaller than in our VDF, also verification time is about this factor smaller. A drawback of his construction is that the overhead for computing the proof  $\pi$  is larger, though it improved significantly in the latest writeup. It's currently at  $O(T/\log(T))$  multiplications (our construction just needs  $O(\sqrt{T} \cdot \log(T))$ ). The computation of the proof can be parallelized to some extent in both constructions.

Another potential advantage (communicated to us by Dan Boneh) of our proof system is that it can be applied for any underlying endomorphism, not just the squaring operation. This could be useful to construct new VDFs, potentially achieving post-quantum security, though currently we don't know of any such instantiations.

In summary, Wesolowski's proof system has shorter proofs and faster verification time. Our proof system allows for more efficient computation of the proof, does not require any computational assumptions and seems to apply in a more general setting.

Wesolowski and the survey [4] also discuss how to instantiate those proof systems in other groups than  $\mathbb{Z}_N^*$ , including groups that will not require trusted setup. We'll discuss this in more detail in §6.1, for now let us just mention that if instantiated in such groups the proof systems will rely on a computational assumption (for Wesolowski's construction, in addition the root assumption), which basically states that it must be hard to find group elements of small order.

## 1.6 Outline

In §2 we discuss the slightly different algebraic setting used here as compared to [10]. We then present the protocol in §3 and the security proof in §4. In §5 we define VDFs, and in §6 we discuss how the protocol is turned into a VDF and discuss several efficiency and security issues.

## 1.7 Notation

For a set  $\mathcal{X}$ ,  $x \stackrel{\$}{\leftarrow} \mathcal{X}$  means  $x$  is assigned a random value from  $\mathcal{X}$ . For a randomized algorithm  $\text{alg}$  we denote with  $x \stackrel{\$}{\leftarrow} \text{alg}$  that  $x$  is assigned the output of  $\text{alg}$  on fresh random coins, if  $\text{alg}$  is deterministic we just write  $x \leftarrow \text{alg}$ .

---

<sup>2</sup> This construction appears in the 2nd version of the eprint paper [14] from July 1st and improves over the construction in the first posting.

## 2 The algebraic setting

The exact algebraic setting for our proof system differs slightly from the setting of the original RSW time-lock puzzle [10]. First, we require that  $N = p \cdot q$  is the product of *safe* primes (a prime  $p$  is safe if  $(p - 1)/2$  is also prime), and moreover we perform the computation in the group of *signed* quadratic residues  $QR_N^+$ . In an earlier version of this paper we used “normal” quadratic residues  $QR_N$ .  $QR_N^+$  is isomorphic to  $QR_N$ , but makes the protocol slightly simpler as unlike for  $QR_N$ , one can efficiently decide if an element is in  $QR_N^+$ . Moreover using  $QR_N^+$  instead of  $QR_N$  will make the proof *unique*, i.e., it’s hard to come up with any proof other than the one generated by the honest prover, so we even get a unique PoSW. As working with  $QR_N$  instead of  $QR_N^+$  is more intuitive, in the proof we’ll assume the proof is over  $QR_N$ .

As we’ll outline below, if computing  $x^{2^T}$  is hard in the original RSW setting, it will remain hard in our setting (the other direction is not clear, it might be that our setting is more secure).

### 2.1 Signed quadratic residues

For two safe primes  $p$  and  $q$ , and  $N := p \cdot q$  we denote the quadratic residues with  $QR_N \stackrel{\text{def}}{=} \{z^2 \bmod N : z \in \mathbb{Z}_N^*\}$ , and the signed quadratic residues [6, 7] are defined as the group

$$QR_N^+ \stackrel{\text{def}}{=} \{|x| : x \in QR_N\},$$

where  $|x|$  is the absolute value when representing the elements of  $\mathbb{Z}_N^*$  as  $\{-(N-1)/2, \dots, (N-1)/2\}$ . Since  $-1 \in \mathbb{Z}_N^*$  is a quadratic non-residue with Jacobi symbol  $+1$ , the map  $|\cdot|$  acts as an (efficiently-computable) isomorphism<sup>3</sup> from  $QR_N$  to  $QR_N^+$ , and as a result  $QR_N^+$  is also a cyclic group, with the group operation defined as

$$a \circ b \stackrel{\text{def}}{=} |a \cdot b \bmod N|.$$

However, unlike for  $QR_N$ , membership in  $QR_N^+$  can be efficiently tested since  $QR_N^+ = J_N^+$  where  $J_N$  is the group of elements with Jacobi symbol  $+1$  and

$$J_N^+ \stackrel{\text{def}}{=} \{|x| : x \in J_N\} = J_N / \{\pm 1\}.$$

In other words, to test whether a given  $x \in \mathbb{Z}_N^*$  (represented as  $\{-(N-1)/2, \dots, (N-1)/2\}$ ) belongs also to  $QR_N^+$ , ensure that  $x \geq 0$  and that its Jacobi symbol is  $+1$ .

### 2.2 Using $(QR_N^+, \circ)$ instead $(\mathbb{Z}_N^*, \cdot)$

Recall that the assumption underlying the security of the RSW time-lock puzzle [10] states that computing  $x^{2^T}$  is hard in  $(\mathbb{Z}_N^*, \cdot)$ . We note that using  $(QR_N^+, \circ)$  instead (i.e., when  $x \in QR_N^+$  and squaring is defined as  $x^2 \stackrel{\text{def}}{=} x \circ x$ ), as we require for our protocol, will not make this assumption any weaker. By the two reductions below, we’ll lose at most a factor  $4 \cdot 2 = 8$  in advantage.

First, let us observe that using  $(QR_N, \cdot)$  instead of  $(\mathbb{Z}_N^*, \cdot)$  can only make the problem harder: Because  $|QR_N| = |\mathbb{Z}_N^*|/4$ , a random element in  $\mathbb{Z}_N^*$  also belongs to  $QR_N$  with probability  $1/4$ . So if one can break the assumption with probability  $\epsilon$  over  $QR_N$ , we still can break it with probability  $\epsilon/4$  over  $\mathbb{Z}_N^*$ .

<sup>3</sup> Note, however, that the inverse of this isomorphism is hard to compute under the quadratic residuosity assumption.

Second, we observe that using  $(QR_N^+, \circ)$  instead of  $(QR_N, \cdot)$  will not make computing  $x^{2^T}$  significantly easier: Consider any  $x \in QR_N$  and let  $y := x^{2^T} \bmod N$  in  $(QR_N, \cdot)$ , and let  $x' = |x|$  and  $y' := x'^{2^T}$  in  $(QR_N^+, \circ)$ , as the groups are isomorphic,  $y' = |y|$ , so  $y = |y'|^{-1}$ , which means  $y \in \{y', N - y'\}$ . Although we can't efficiently decide if  $y = y'$  or  $y = N - y'$  (as it would contradict the quadratic residuosity assumption), we can pick one of the two values at random and will get the right one with probability  $1/2$ . This shows that given an algorithm that finds  $x^{2^T}$  in  $QR_N^+$  in time  $t$  with probability  $\delta$ , we get an algorithm that computes  $x^{2^T}$  in  $QR_N^+$  in basically the same time  $t$  and probability  $\delta/2$ .

### 2.3 On using safe primes

Another difference to the setting of [10] is that we assume that  $N = p \cdot q$  is the product of random *safe* primes, whereas [10] just assume random primes. We do this to make sure that  $QR_N$  (and thus also  $QR_N^+$ ) contains no sub-group of small order, this property is required to prove statistical soundness.

It is conjectured that for some constant  $c$ , there are  $c \cdot 2^\lambda/n^2$  safe  $\lambda$ -bit primes (cf. [13]), so a random  $n$  bit prime is safe with probability  $\approx c/n$ . Under this assumption, the product of two random  $n$ -bit primes will be the product of two safe primes with probability  $c^2/n^2$ .

## 3 The protocol

Our protocol, where  $\mathcal{P}$  convinces  $\mathcal{V}$  it solved an RSW puzzle, goes as follows:

- The verifier  $\mathcal{V}$  and prover  $\mathcal{P}$  have as common input an RSW puzzle  $(N, x, T)$  and a statistical security parameter  $\lambda$ . Here  $T \in \mathbb{N}$ ,  $N = p \cdot q$  is the product of safe primes and  $x \in QR_N^+$ .
- $\mathcal{P}$  solves the puzzle by computing  $y = x^{2^T}$  (making  $T$  sequential squarings in  $(QR_N^+, \circ)$ ), and sends  $y$  to  $\mathcal{V}$ .
- Now  $\mathcal{P}$  and  $\mathcal{V}$  iterate the “halving protocol” below. In this subprotocol, on common input  $(N, x, T, y)$  the output is either of the form  $(N, x', \lceil T/2 \rceil, y')$ , in which case it is used as input to the next iteration of the halving subprotocol, or the protocol has stopped with verifier output in  $\{\text{reject}, \text{accept}\}$ .

### 3.1 The halving subprotocol

On common input  $(N, x, T, y)$

1. If  $T = 1$  then  $\mathcal{V}$  outputs **accept** if  $y = x^{2^T} = x^2$  and **reject** otherwise. If  $T > 1$  go to the next step.
2. The prover  $\mathcal{P}$  sends  $\mu = x^{2^{T/2}}$  to  $\mathcal{V}$ .
3. If  $\mu \notin QR_N^+$  then  $\mathcal{V}$  outputs **reject**, otherwise  $\mathcal{V}$  samples a random  $r \xleftarrow{\$} \mathbb{Z}_{2^\lambda}$  and sends it to  $\mathcal{P}$ .
4. If  $T/2$  is even,  $\mathcal{P}$  and  $\mathcal{V}$  output

$$(N, x', T/2, y')$$

where

$$x' := x^r \cdot \mu \quad (= x^{r+2^{T/2}})$$

$$y' := \mu^r \cdot y \quad (= x^{r \cdot 2^{T/2} + 2^T})$$

(note that if  $y = x^{2^T}$  then  $y' = x'^{2^{T/2}}$ ). If  $T/2$  is odd, output

$$(N, x', (T+1)/2, y'^2).$$

### 3.2 Security statement

► **Theorem 1.** *If the input  $(N, x, T)$  to the protocol satisfies*

1.  $N = p \cdot q$  is the product of safe primes, i.e.,  $p = 2p' + 1, q = 2q' + 1$  for primes  $p', q'$ .
2.  $\langle x \rangle = QR_N^+$ .<sup>4</sup>
3.  $2^\lambda \leq \min\{p', q'\}$

*Then for any malicious prover  $\tilde{\mathcal{P}}$  who sends as first message  $y$  anything else than the solution to the RSW time-lock puzzle, i.e.,*

$$y \neq x^{2^T}$$

*$\mathcal{V}$  will finally output `accept` with probability at most*

$$\frac{3 \log(T)}{2^\lambda}.$$

## 4 Security proof

### 4.1 Usage of $QR_N$ instead $QR_N^+$ in the proof

We'll prove Theorem 1 where the signed quadratic residues  $(QR_N^+, \circ)$  are replaced with regular quadratic residues  $(QR_N, \cdot)$  throughout. This will make the proof a bit more intuitive as multiplication modulo  $N$  as in  $QR_N$  is a more familiar and simpler operation than the  $\circ$  operation in  $QR_N^+$  (which additionally requires the  $|\cdot|$  mapping after each multiplication). As discussed in §2, those two groups are isomorphic, so the proof for  $(QR_N, \cdot)$  implies the same security for  $(QR_N^+, \circ)$ .

The main reason we don't use  $(QR_N, \cdot)$  in the actual protocol is only because in step 3. of the halving subprotocol  $\mathcal{V}$  needs to check if  $\mu \in QR_N^+$ , which would not be efficient if we used  $QR_N$  (in an earlier version of the protocol we did use  $QR_N$ , and  $\mathcal{P}$  had to send  $\mu'$  s.t.  $\mu'^2 = \mu$ , the verifier would then compute  $\mu := \mu'^2$  can thus could be sure that  $\mu \in QR_N$ . As here  $\mathcal{P}$  can send any of the 4 roots of  $\mu$ , this protocol was not unique).

### 4.2 The language $\mathcal{L}$

It will be convenient to define the language

$$\mathcal{L} = \{(N, x, T, y) \quad : \quad y \neq x^{2^T} \pmod N \text{ and } \langle x \rangle = QR_N\}$$

We'll establish the following lemma.

► **Lemma 2.** *For  $N, \lambda$  as in Thm. 1, and any malicious prover  $\tilde{\mathcal{P}}$  the following holds. If the input to the halving protocol in §3.1 satisfies*

$$(N, x, T, y) \in \mathcal{L}$$

*then with probability  $\geq 1 - 3/2^\lambda$  (over the choice of  $r$ )  $\mathcal{V}$ 's output is either `reject` or satisfies*

$$(N, x', \lceil T/2 \rceil, y') \in \mathcal{L}$$

Before we prove the lemma, let's see how it implies Theorem 1.

<sup>4</sup> That is,  $x$  generates  $QR_N^+$ , the quadratic residues modulo  $N$ . For our choice of  $N$  we have  $|QR_N^+| = |QR_N| = p'q'$ , so  $\langle x \rangle \stackrel{\text{def}}{=} \{x, x^2, \dots, x^{p'q'}\} = QR_N^+$ .

**Proof of Theorem 1.** In every iteration of the halving protocol the time parameter decreases from  $T$  to  $\lceil T/2 \rceil$  and it stops once  $T = 1$ , this means we iterate for at most  $\lceil \log(T) \rceil$  rounds. By assumption, the input  $(N, x, T, y)$  to the first iteration is in  $\mathcal{L}$ , and by construction, the only case where  $\mathcal{V}$  outputs **accept** is on an input  $(N, x, 1, y)$  where  $y = x^{2^T} = x^2 \pmod N$ , in particular, this input is *not* in  $\mathcal{L}$ .

So, if  $\mathcal{V}$  outputs **accept**, there must be one iteration of the halving protocol where the input is in  $\mathcal{L}$  but the output is not. By Lemma 2, for any particular iteration this happens with probability  $\leq 3/2^\lambda$ . By the union bound, the probability of this happening in any of the  $\lceil \log(T) \rceil - 1$  rounds can be upper bounded by  $3 \log(T)/2^\lambda$  as claimed.  $\blacktriangleleft$

**Proof of Lemma 2.** We just consider the case where  $T$  is even, the odd  $T$  case is almost identical.

Assuming the input to the halving protocol satisfies  $(N, x, T, y) \in \mathcal{L}$ , we must bound the probability that  $\mathcal{V}$  outputs **reject** or the output  $(N, x', T/2, y') \notin \mathcal{L}$ .

If  $T = 1$  then  $\mathcal{V}$  outputs **reject** and we're done. Otherwise, if  $\tilde{\mathcal{P}}$  sends a  $\mu \notin QR_N$  in step 2. then  $\mathcal{V}$  outputs **reject** in step 3. and we're done. So from now we assume  $\mu \in QR_N$ . We must bound

$$\Pr_r[(y' = x'^{2^{T/2}}) \vee (\langle x' \rangle \neq QR_N)] \leq 3/2^\lambda$$

using  $\Pr[a \vee b] = \Pr[a \wedge \bar{b}] + \Pr[b]$  we rewrite this as

$$\Pr_r[y' = x'^{2^{T/2}} \wedge \langle x' \rangle = QR_N] + \Pr_r[\langle x' \rangle \neq QR_N] \leq 3/2^\lambda \quad (3)$$

Eq. (3) follows by the two claims below.

► **Claim 1.**  $\Pr_r[\langle x' \rangle \neq QR_N] \leq 2/2^\lambda$ .

**Proof of Claim.** We'll denote with  $e_\mu$  the unique value in  $\mathbb{Z}_{p'q'}$  satisfying  $x^{e_\mu} = \mu$  (it's unique as  $\mu \in \langle x \rangle = QR_N$  and  $|QR_N| = p'q'$ ). As  $x, \mu \in QR_N$ , also  $x' = x^r \cdot \mu = x^{r+e_\mu}$  is in  $QR_N$ , and  $\langle x' \rangle = QR_N$  holds if  $\text{ord}(x') = p'q'$ , which is the case except if  $(r+e_\mu) = 0 \pmod{p'}$  or  $(r+e_\mu) = 0 \pmod{q'}$  or equivalently (using that  $2^\lambda < \min(p', q')$ ) if

$$r \in \mathcal{B} \stackrel{\text{def}}{=} \{\mathbb{Z}_{2^\lambda} \cap \{(-e_\mu \pmod{p'}), (-e_\mu \pmod{q'})\}\}. \quad (4)$$

Clearly  $|\mathcal{B}| \leq 2$  and the claim follows.  $\blacktriangleleft$

► **Claim 2.**  $\Pr_r[y' = x'^{2^{T/2}} \pmod N \wedge \langle x' \rangle = QR_N] \leq 1/2^\lambda$ .

**Proof of Claim.** If  $y \notin QR_N$ , then also  $y' = \mu^r \cdot y \notin QR_N$  (as  $a \in QR_N, b \notin QR_N$  implies  $a \cdot b \notin QR_N$ ). As  $\langle x' \rangle = QR_N$  and  $y' \neq x'^{2^{T/2}}$  can't hold simultaneously in this case the probability in the claim is 0. From now on we consider the case  $y \in QR_N$ . We have

$$\begin{aligned} \Pr_r[y' = x'^{2^{T/2}} \wedge \langle x' \rangle = QR_N] &= \\ \Pr_r[y' = x'^{2^{T/2}} \mid \langle x' \rangle = QR_N] \cdot \Pr_r[\langle x' \rangle = QR_N] & \end{aligned} \quad (5)$$

For the second factor in (5) we have with  $\mathcal{B}$  as in (4)

$$\Pr_r[\langle x' \rangle = QR_N] = \frac{2^\lambda - |\mathcal{B}|}{2^\lambda}. \quad (6)$$

Conditioned on  $\langle x' \rangle = QR_N$  the  $r$  is uniform in  $\mathbb{Z}_{2^\lambda} \setminus \mathcal{B}$ , so the first factor in (5) is

$$\Pr_r[y' = x'^{2^{T/2}} \mid \langle x' \rangle = QR_N] = \Pr_{r \in \mathbb{Z}_{2^\lambda} \setminus \mathcal{B}}[y' = x'^{2^{T/2}}]. \quad (7)$$

Let  $e_y \in \mathbb{Z}_{p'q'}$  be the unique value such that  $x^{e_y} = y$ . Using  $\langle x \rangle = QR_N$  in the last step below we can rewrite

$$\begin{aligned} y' &= x'^{2^{T/2}} \pmod{N} && \iff \\ \mu^r y &= (x^r \mu)^{2^{T/2}} \pmod{N} && \iff \\ x^{r \cdot e_\mu + e_y} &= x^{(r+e_\mu) \cdot 2^{T/2}} \pmod{N} && \iff \\ r \cdot e_\mu + e_y &= (r + e_\mu) \cdot 2^{T/2} \pmod{p'q'} \end{aligned}$$

rearranging terms

$$r(e_\mu - 2^{T/2}) + e_y - e_\mu 2^{T/2} = 0 \pmod{p'q'}. \quad (8)$$

If  $e_\mu = 2^{T/2}$  this becomes

$$e_y - 2^T = 0 \pmod{p'q'}$$

which does not hold as by assumption we have  $y \neq x^{2^T}$ . So from now on we assume  $e_\mu \neq 2^{T/2} \pmod{p'q'}$ . Then for  $a = e_\mu - 2^{T/2} \neq 0 \pmod{p'q'}$  (and  $b = e_y - e_\mu 2^{T/2}$ ) eq. (8) becomes

$$r \cdot a = b \pmod{p'q'}$$

which holds for at most one choice of  $r$  from its domain  $\mathbb{Z}_{2^\lambda} \setminus \mathcal{B}$ , thus

$$\Pr_{r \in \mathbb{Z}_{2^\lambda} \setminus \mathcal{B}}[y' = x'^{2^{T/2}}] \leq \frac{1}{2^\lambda - |\mathcal{B}|}$$

and the claim follows from the above equation and (5)-(7) as

$$\begin{aligned} \Pr_r[y' = x'^{2^{T/2}} \wedge \langle x' \rangle = QR_N] &= \\ \Pr_{r \in \mathbb{Z}_{2^\lambda} \setminus \mathcal{B}}[y' = x'^{2^{T/2}}] \cdot \Pr_r[\langle x' \rangle = QR_N] &\leq \frac{1}{2^\lambda - |\mathcal{B}|} \cdot \frac{2^\lambda - |\mathcal{B}|}{2^\lambda} \leq \frac{1}{2^\lambda}. \end{aligned} \quad \blacktriangleleft$$

## 5 Verifiable delay functions

In this section we define verifiable delay functions (VDF) mostly following the definition from [3]. A VDF is defined by a four-tuple of algorithms:

**VDF.Setup**( $1^\lambda$ )  $\rightarrow$  **pp** on input a statistical security parameter  $1^\lambda$  outputs public parameters **pp**.

**VDF.Gen**(**pp**,  $T$ )  $\rightarrow$   $(x, T)$  on input a time parameter  $T \in \mathbb{N}$ , samples an input  $x$ .

**VDF.Sol**(**pp**,  $(x, T)$ )  $\rightarrow$   $(y, \pi)$  on input  $(x, T)$  outputs  $(y, \pi)$ , where  $\pi$  is a proof that the output  $y$  has been correctly computed.

**VDF.Ver**(**pp**,  $(x, T)$ ,  $(y, \pi)$ )  $\rightarrow$  {**accept/reject**} given an input/output tuple  $(x, T)$ ,  $(y, \pi)$  outputs either **accept** or **reject**.

The **VDF.Setup** and **VDF.Gen** algorithms are probabilistic, **VDF.Sol** and **VDF.Ver** are deterministic. They all run in time  $\text{poly}(\log(T), \lambda)$ .



## 5.1 The statistical security parameter

$\lambda$  measures the bit-security we expect from our protocol, i.e., an adversary of complexity  $\tau$  should have advantage no more than  $\approx \tau/2^\lambda$  in breaking the scheme. It only makes sense to consider time parameters  $T$  that are much smaller than  $2^\lambda$  (say we require  $T \leq 2^{\lambda/2}$ ) so the sequential running time of the honest prover is much smaller than what is required to break the underlying hardness assumptions.

## 5.2 Efficiency of solving

The  $\text{VDF.Sol}$  algorithm can compute the output  $y$  in  $T$  sequential steps (in this work a “sequential step” is the  $\circ$  operation, which basically is a multiplication modulo  $N$ ). Moreover we require that  $\pi$  can be computed with in much fewer than  $T$  steps. As we’ll discuss in §6.2, we’ll achieve  $O(\sqrt{T} \log(T))$  sequential steps, and less if parallelism is available. In [3] the requirement is more relaxed, they compute  $\pi$  in parallel with  $y$  using bounded  $\text{poly}(\log(T), \lambda)$  parallelism, so the  $\pi$  is available shortly after  $y$  is computed, but overall the computation is much larger than  $T$ . As discussed in the introduction, Wesolowski’s VDF [14] requires  $O(T/\log(T))$  steps to compute  $\pi$ .

## 5.3 Completeness

The completeness property simply requires that correctly generated proofs will always accept, that is, for any  $\lambda, T$

$$\Pr \left[ \begin{array}{l} \text{VDF.Ver}(\mathbf{pp}, (x, T), (y, \pi)) = \text{accept} \\ \text{where} \\ \mathbf{pp} \xleftarrow{\$} \text{VDF.Setup}(1^\lambda) \\ (x, T) \xleftarrow{\$} \text{VDF.Gen}(\mathbf{pp}, T) \\ (y, \pi) \leftarrow \text{VDF.Sol}(\mathbf{pp}, (x, T)) \end{array} \right] = 1$$

## 5.4 Security (sequentiality)

The first security property is sequentiality. For this we consider a two part adversary  $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ , where  $\mathcal{A}_1$  can run a pre-computation and choose  $T$ . Then  $\mathcal{A}_2$  gets a random challenge for time  $T$  together with the output **state** of the precomputation, we require that whenever

$$\Pr \left[ \begin{array}{l} \text{VDF.Ver}(\mathbf{pp}, (x, T), (\tilde{y}, \tilde{\pi})) = \text{accept} \\ \text{where} \\ \mathbf{pp} \xleftarrow{\$} \text{VDF.Setup}(1^\lambda) \\ (T, \text{state}) \xleftarrow{\$} \mathcal{A}_1(\mathbf{pp}) \\ (x, T) \xleftarrow{\$} \text{VDF.Gen}(\mathbf{pp}, T) \\ (\tilde{y}, \tilde{\pi}) \xleftarrow{\$} \mathcal{A}_2(\mathbf{pp}, (x, T), \text{state}) \end{array} \right] \neq \text{negl}(\lambda)$$

the  $\mathcal{A}_2$  adversary must use almost the same *sequential* time  $T$  as required by an honest execution of  $\text{VDF.Sol}(\mathbf{pp}, (\pi, T))$ , and this even holds if  $\mathcal{A}$  is allowed massive parallel computation (say we just bound the total computation to  $2^{\lambda/2}$ ). This means there’s no possible speedup to compute the VDF output by using parallelism. Let us stress that by this we mean any parallelism that goes beyond what can be used to speed up a single sequential step, which here is a multiplication in  $\mathbb{Z}_N^*$ , and we assume the honest prover can use such bounded parallelism.

## 5.5 Security (soundness)

The second security property is soundness, which means that one cannot come up with an accepting proof  $\tilde{\pi}$  for a wrong statement. Formally, for an adversary  $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$  we have (unlike in the previous definition, here we don't make any assumption about  $\mathcal{A}_2$ 's sequential running time, just the total running time of  $\mathcal{A}$  must be bounded to, say  $2^{\lambda/2}$ )

$$\Pr \left[ \begin{array}{l} \text{VDF.Ver}(\mathbf{pp}, (x, T), (\tilde{y}, \tilde{\pi})) = \text{accept} \\ \text{and } \tilde{y} \neq y \\ \text{where} \\ \mathbf{pp} \stackrel{\$}{\leftarrow} \text{VDF.Setup}(1^\lambda) \\ (T, \text{state}) \stackrel{\$}{\leftarrow} \mathcal{A}_1(\mathbf{pp}) \\ (x, T) \stackrel{\$}{\leftarrow} \text{VDF.Gen}(\mathbf{pp}, T) \\ (y, \pi) \leftarrow \text{VDF.Sol}(\mathbf{pp}, (x, T)) \\ (\tilde{y}, \tilde{\pi}) \stackrel{\$}{\leftarrow} \mathcal{A}_2(\mathbf{pp}, (x, T), \text{state}) \end{array} \right] = \text{negl}(\lambda)$$

## 6 A VDF from RSW

In this section we explain the simple transformation of the protocol from §3 into a VDF and then discuss the efficiency, security and some other issues of this construction.

To keep things simple we'll assume that the time parameter  $T = 2^t$  is a power of two. The four algorithms from §5 are instantiated as

**VDF.Setup**( $1^\lambda$ ) The statistical security parameter  $\lambda$  defines another security parameter  $\lambda_{\text{RSA}}$  specifying the bitlength of an RSA modulus, where  $\lambda_{\text{RSA}}$  should be at least as large so that an  $\lambda_{\text{RSA}}$  bit RSA modulus offers  $\lambda$  bits of security (e.g.  $\lambda = 100$  and  $\lambda_{\text{RSA}} = 2048$ ). As hardness of factoring is subexponential, while the soundness of our protocol is exponentially small in  $\lambda$  (in the random oracle model), we can without loss of generality assume that  $\lambda \leq \lambda_{\text{RSA}}/2$ , so point 3. in the statement of Theorem 1 is satisfied.

The setup algorithm samples two random  $\lambda_{\text{RSA}}/2$  bit safe primes  $p, q$  and outputs as public parameters the single  $\lambda_{\text{RSA}}$  bit RSA modulus  $N := p \cdot q$ .

**VDF.Gen**( $N, T$ ) samples a random  $x \in QR_N^+$  and outputs  $(x, T)$ .

**VDF.Sol**( $N, (x, T)$ ) outputs  $(y, \pi)$  where  $y = x^{2^T}$  is the solution of the RSW time-lock puzzle (but over  $(QR_N^+, \circ)$  not  $(\mathbb{Z}_N^*, \cdot)$ ) and  $\pi = \{\mu_i\}_{i \in [t]}$  is a proof that  $y$  has been correctly computed. It is derived by applying the Fiat-Shamir heuristic to the protocol in §3. Recall that in this heuristic the public-coin challenges  $r_i \in \mathbb{Z}_{2^\lambda}$  of the verifier are replaced with a hash of the last prover message. Concretely, we fix a hash function  $\text{hash} : \mathbb{Z} \times \mathbb{Z}_N^4 \rightarrow \mathbb{Z}_{2^\lambda}$ , let  $(x_1, y_1) := (x, y)$  and for  $i = 1 \dots t$  let<sup>5</sup>

<sup>5</sup> Note that in the Fiat-Shamir heuristic, we not just hash the first prover message  $\mu_i$  in eq. (9), but also the statement  $(x_i, T/2^{i-1}, y_i)$  of the halving subprotocol. It has been observed that this is necessary in a setting like ours, where the prover has some influence on the statement to be proven [1]. There exists an easy attack (communicated to us by Benjamin Wesolowski) on uniqueness of the VDF if the  $y$ 's are not included in the hash: for  $(x, T) = (x_1, T_1)$ , pick  $\mu_1$  at random, let  $r_1 = \text{hash}((x, T), \mu_1)$ ,  $y = (x^{2^{T/2}} / \mu_1)^{r_1} \mu_1^{2^{T/2}}$ ,  $x_2 = x_1^{r_1} \mu_1$ ,  $y_2 = \mu_1^{r_1} y$ . The above  $y$  is almost certainly wrong, i.e.,  $y \neq x^{2^T}$ , but by construction  $y_2 = x_2^{2^{T/2}}$ , so one can continue with the honest proof.

$$\begin{aligned}
\mu_i &:= x_i^{2^{T/2^i}} \in QR_N^+ \\
r_i &:= \text{hash}((x_i, T/2^{i-1}, y_i), \mu_i) \in \mathbb{Z}_{2^\lambda} \\
x_{i+1} &:= x_i^{r_i} \circ \mu_i \\
y_{i+1} &:= \mu_i^{r_i} \circ y_i
\end{aligned} \tag{9}$$

**VDF.Ver**( $N, (x, T), (y, \pi)$ ) parses  $\pi = \{\mu_i\}_{i \in [t]}$  and checks if  $x, y$  and all  $\mu_i$  are in  $QR_N^+$ , if this is not the case output **reject**. Otherwise set  $(x_1, y_1) := (x, y)$  and then for  $i = 1 \dots t$  compute

$$r_i := \text{hash}((x_i, T/2^{i-1}, y_i), \mu_i) \tag{10}$$

$$x_{i+1} := x_i^{r_i} \circ \mu_i \tag{10}$$

$$y_{i+1} := \mu_i^{r_i} \circ y_i \tag{11}$$

Finally check whether

$$y_{t+1} \stackrel{?}{=} x_{t+1}^2 \tag{12}$$

and output **accept** if this holds, otherwise output **reject**.

## 6.1 Public parameters for the VDF

For the security of the VDF it's crucial that a prover does not know the factorization of the public parameter  $N$ , as otherwise he could compute  $x^{2^T}$  in just two exponentiations as in eq. (1). Thus one either has to rely on a trusted party, or use multiparty-computation to sample  $N$ . In particular, it's possible to sample  $N$  securely as long as not all the participants in the multiparty computation are malicious. Such an ‘‘MPC ceremony’’ has been done before, e.g. to set up the common random string for Zcash.<sup>6</sup> This is in contrast to the random-oracle based PoSW [9, 5] which don't require a setup procedure at all.

To avoid trusted setup, Boneh et al. [4] and Wesolowski [14] suggested to use class groups of an imaginary quadratic field [2] instead of an RSA group. Recall that the statistical soundness of our proof systems relies on the fact that the underlying group (the quadratic residues of  $\mathbb{Z}_N^*$  where  $N$  is the product of safe primes) has no subgroups of small order. If the underlying group does have groups of small order, then *computational* soundness holds under the assumption that it's hard to find elements of small order, which is conjectured to hold for the class groups mentioned above.

## 6.2 Efficiency of the VDF

### 6.2.1 Cost of verification

The cost of running the verification  $\text{VDF.Ver}(N, (x, T = 2^t), (y, \pi))$  is dominated by the  $2t$  exponentiations (with  $\lambda$  bit long exponents) in eq. (10-11). As exponentiation with a random  $\lambda$  bit exponent costs about  $1.5\lambda$  multiplications,<sup>7</sup> the cost of verification is around  $3 \cdot \lambda \cdot t$

<sup>6</sup> <https://z.cash/technology/paramgen.html>

<sup>7</sup> Exponentiation is typically done via ‘‘square and multiply’’, which for a  $z$  bit exponent with hamming weight  $h(z)$  requires  $z + h(z)$  multiplications, or about  $1.5 \cdot z$  multiplication for a random exponent (where  $h(z) \approx z/2$ ).

multiplications.<sup>8</sup> For concreteness, consider an implementation where  $\lambda = 100$ ,  $\lambda_{\text{RSA}} = 2048$  and assume  $t = 40$ , this gives a cost of about  $3 \cdot \lambda \cdot t = 12000$  multiplications, which corresponds to  $12000/(2048 \cdot 1.5) \approx 4$  full exponentiations in  $\mathbb{Z}_N^*$ .

## 6.2.2 A minor efficiency improvement

There's a simple way to save on verification time and proof size. Currently, for  $T = 2^t$  we run the halving protocol for  $t$  rounds, and then in eq. (12) check if  $y_{t+1} \stackrel{?}{=} x_{t+1}^2$ . For any integer  $\Delta \geq 0$  we could run the protocol for just  $t - \Delta$  rounds, but then the verifier must check if  $y_{t+1-\Delta} \stackrel{?}{=} x_{t+1-\Delta}^{2^{2^\Delta}}$ , which requires  $2^\Delta$  squarings (more generally, if  $T$  is not a power of 2 then the check becomes  $y_{t+1-\Delta} \stackrel{?}{=} x_{t+1-\Delta}^{2^{T_{t+1-\Delta}}} \pmod N$  where  $T_1 = T, T_i = \lceil T_{i-1}/2 \rceil$ ).

If we set, say  $\Delta = 10$ , this saves 10 rounds and thus reduces the proof size by 25% from 40 to 30 elements. The verification time decreases by around 15% (we save 20 short exponentiations as in eq. (10,11) at the price of 1024 extra squarings).

## 6.2.3 Cost of computing the proof

Computing the proof  $(y, \pi) \leftarrow \text{VDF.Sol}(N, (x, T))$  requires one to solve the underlying RSW puzzle  $y = x^{2^T}$ , which is done by squaring  $x$  sequentially  $T$  times (the security of the RSW puzzle and thus also our VDF relies on the assumption that there's no shortcut to this computation).

On top of that, for the VDF we also must compute the proof  $\pi = \{\mu_i\}_{i \in [t]}$  where  $\mu_i = x_i^{2^{T/2^i}}$ . But we still assume that  $T = 2^t$  is a power of 2.

If naïvely implemented, computing the  $\mu_i$  will require  $T/2$  squarings for  $\mu_1$ ,  $T/4$  for  $\mu_2$  etc., adding up to a total of  $T \approx T/2 + T/4 + T/8 \dots + 1$  sequential steps. Fortunately we don't have to compute  $\mu_1 = x^{2^{T/2}}$  as we already did so while computing  $y = x^{2^T}$  by repeated squaring (cf. eq. (2)). This observation already saves us half the overhead. We can also compute the remaining  $\mu_2, \mu_3, \dots$  using stored values, but it becomes increasingly costly, as we discuss below.

In general, for some  $s \in [t]$  the prover can compute  $\mu_1, \dots, \mu_s$  using stored values, and then fully recompute the remaining  $\mu_{s+1} = x_{s+1}^{2^{T/2^{s+1}}}, \mu_{s+2}, \dots, \mu_t$  which will only require  $T/2^{s+1} + T/2^{s+2} \dots < T/2^s$  squarings.

To see how the  $\mu_i$ 's can be efficiently computed for small  $i$ , for  $z \in QR_N^+$  let  $\bar{z}$  denote  $z$ 's log to basis  $x$ , i.e.,  $x^{\bar{z}} = z$ . We have  $\bar{x}_1 = 1, \bar{y}_1 = 2^T$  and

$$\begin{aligned} \bar{\mu}_i &:= \bar{x}_i \cdot 2^{T/2^i} \\ \bar{x}_{i+1} &:= r_i \cdot \bar{x}_i + \bar{\mu}_i \\ \bar{y}_{i+1} &:= r_i \cdot \bar{\mu}_i + \bar{y}_i \end{aligned}$$

How those exponents concretely develop for  $i = 1$  to 3 is illustrated in Figure 1. For example, we can compute  $\mu_3$  assuming we stored the  $x^{2^{T/8}}, x^{2^{T^3/8}}, x^{2^{T^5/8}}, x^{2^{T^7/8}}$  values as

$$\mu_3 = (x^{2^{T/8}})^{r_1 \cdot r_2} \cdot (x^{2^{T^5/8}})^{r_2} \cdot (x^{2^{T^3/8}})^{r_1} \cdot x^{2^{T^7/8}}$$

<sup>8</sup> Here multiplication means the  $\circ$  operation, which requires one multiplication modulo  $N$ , followed by the map  $|\cdot|$ . As the cost of this map is marginal compared to the multiplication we just ignore it.

## 60:14 Simple Verifiable Delay Functions

$i$	$\bar{x}_i$	$\bar{\mu}_i$	$\bar{y}_i$
1	1	$2^{t/2}$	$2^t$
2	$r_1 + 2^{t/2}$	$r_1 \cdot 2^{t/4} + 2^{3t/4}$	$r_1 \cdot 2^{t/2} + 2^t$
3	$r_1 \cdot r_2 + r_2 \cdot 2^{t/2} + 2^{t/4} \cdot r_1 + 2^{3t/4}$	$r_1 \cdot r_2 \cdot 2^{t/8} + r_2 \cdot 2^{5t/8} + r_1 \cdot 2^{3t/8} + 2^{7t/8}$	$r_1 \cdot r_2 \cdot 2^{t/4} + r_2 \cdot 2^{3t/4} + r_1 \cdot 2^{t/2} + 2^t$
$\vdots$	$\vdots$	$\vdots$	$\vdots$

■ **Figure 1** Exponents of the the values in the protocol, here  $z = x^z$ .

In general, computing  $\mu_1, \dots, \mu_s$  will require to store  $2^s$  values  $\{x^{2^{T-i/2^s}}\}_{i \in [2^s]}$ , and then compute  $2^s$  exponentiations with exponents of bitlength at most  $\lambda \cdot (s-1)$  (and half that on average). We can't speed this up by first taking the exponents modulo the group order  $p'q'$  as it is not known, but if we have bounded parallelism  $2^p, p \leq (s-2)$  this can be done in  $2^{s-p} \cdot \lambda \cdot (s-1) \cdot \frac{3}{4}$  sequential steps. Summing up, with sufficient space to store  $2^s$  elements in  $\mathbb{Z}_N$  and  $2^p \leq 2^s$  parallelism the proof  $\pi$  can be computed in roughly

$$2^{s-p} \cdot \lambda \cdot (s-1) \cdot \frac{3}{4} + 2^{t-s} \text{ sequential steps and } 2^s \cdot \log(N) \text{ bits of storage}$$

after  $y$  has been computed. For example with a single core  $p = 0$  and  $s = t/2 - \log(t \cdot \lambda)/2$  the number of steps (i.e.,  $\circ$  operations) becomes

$$2^{t/2 - \log(t \cdot \lambda)/2} \cdot \lambda \cdot (s-1) \cdot \frac{3}{4} + 2^{t/2 + \log(t \cdot \lambda)/2} = 2^{t/2} \left( \frac{\lambda(s-1)}{\sqrt{t\lambda}} \cdot \frac{3}{4} + \sqrt{t\lambda} \right) < \sqrt{T} \cdot \frac{11}{8} \cdot \sqrt{\log(T) \cdot \lambda}$$

For our typical values  $t = 40, \lambda = 100$  this is  $\leq 2^{27}$ , and thus over  $2^{40-27} = 2^{13}$  times faster than computing  $y$ , e.g. if computing  $y$  takes  $1h$ , computing  $\pi$  just takes half a second on top. The memory required (to store intermediate values) is around  $2^s \cdot \log N = 2^{t/2 - \log(t \cdot \lambda)/2} \cdot 1024 \leq 2^{27}$  bits, or  $8MB$ .

## 6.3 Security of the VDF

### 6.3.1 Soundness

If we model *hash* as a random oracle, then by Lemma 2 (which is used in the proof of Theorem 1) we are guaranteed that a malicious prover will not find an accepting proof  $(\tilde{y}, \tilde{\pi})$  for a wrong statement  $\tilde{y} \neq x^{2^T}$  except with exponentially small probability. We can even let the malicious prover choose the challenge  $(x, T)$  for which it must forge such a proof itself, the only restriction being that  $x$  must be a generator  $\langle x \rangle = QR_N^+$  (a random  $x \in QR_N^+$  satisfies this almost certainly, but we can't efficiently verify if a given  $x$  is such a generator).

The well known Fiat-Shamir heuristic states that replacing the prover's queries with the output of a random oracle in a sound public-coin interactive proof system results in a sound non-interactive proof system, but this only applies for protocols with a constant number of rounds.

Even though our protocol has logarithmically many rounds, we can directly conclude that our non-interactive proof is sound as follows: if we are given a valid proof for a wrong statement, then, during the execution of the verification algorithm for this proof, we must make a query  $hash(x_i, T/2^{i-1}, y_i, \mu_i)$  where  $(N, x_i, T/2^{i-1}, y_i) \in \mathcal{L}$  ( $\mathcal{L}$  as defined in §4) but for the next query made  $hash(x_{i+1}, T/2^i, y_{i+1}, \mu_{i+1})$  we have  $(N, x_{i+1}, T/2^i, y_{i+1}) \notin \mathcal{L}$ . By Lemma 2, every random oracle query will correspond to such a query with probability at most  $3/2^\lambda$ . Thus, by the union bound, the probability that a malicious prover that makes up to  $q$  queries to *hash* will find such a query (which as outlined is necessary to find an accepting proof for a wrong statement) is at most  $q \cdot 3/2^\lambda$ .

### 6.3.2 Sequentiality

To break sequentiality means computing  $y$  faster than in  $T$  sequential computations. We rely on the same assumption as [10], which simply states that such a shortcut does not exist. As outlined in §2, the fact that we work over  $(QR_N^+, \circ)$  not  $(\mathbb{Z}_N, \cdot)$  only makes the assumption on which we rely weaker, and the fact that in our case  $N$  is the product of *safe* primes doesn't affect the assumption assuming that safe primes are not too sparse.

---

#### References

- 1 David Bernhard, Olivier Pereira, and Bogdan Warinschi. How Not to Prove Yourself: Pitfalls of the Fiat-Shamir Heuristic and Applications to Helios. In Xiaoyun Wang and Kazue Sako, editors, *ASIACRYPT 2012*, volume 7658 of *LNCS*, pages 626–643. Springer, Heidelberg, December 2012. doi:10.1007/978-3-642-34961-4\_38.
- 2 Ingrid Biehl, Johannes A. Buchmann, Safuat Hamdy, and Andreas Meyer. A Signature Scheme Based on the Intractability of Computing Roots. *Des. Codes Cryptography*, 25(3):223–236, 2002.
- 3 Dan Boneh, Joseph Bonneau, Benedikt Bünz, and Ben Fisch. Verifiable Delay Functions. In *CRYPTO 2018*, 2018.
- 4 Dan Boneh, Benedikt Bünz, and Ben Fisch. A Survey of Two Verifiable Delay Functions. Cryptology ePrint Archive, Report 2018/712, 2018. URL: <https://eprint.iacr.org/2018/712>.
- 5 Bram Cohen and Krzysztof Pietrzak. Simple Proofs of Sequential Work. In Jesper Buus Nielsen and Vincent Rijmen, editors, *EUROCRYPT 2018, Part II*, volume 10821 of *LNCS*, pages 451–467. Springer, Heidelberg, April / May 2018. doi:10.1007/978-3-319-78375-8\_15.
- 6 Roger Fischlin and Claus-Peter Schnorr. Stronger Security Proofs for RSA and Rabin Bits. *Journal of Cryptology*, 13(2):221–244, 2000.
- 7 Dennis Hofheinz and Eike Kiltz. The Group of Signed Quadratic Residues and Applications. In Shai Halevi, editor, *CRYPTO 2009*, volume 5677 of *LNCS*, pages 637–653. Springer, Heidelberg, August 2009.
- 8 Carsten Lund, Lance Fortnow, Howard J. Karloff, and Noam Nisan. Algebraic Methods for Interactive Proof Systems. In *31st FOCS*, pages 2–10. IEEE Computer Society Press, October 1990.
- 9 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.
- 10 R. L. Rivest, A. Shamir, and D. A. Wagner. Time-lock Puzzles and Timed-release Crypto. Technical report, Massachusetts Institute of Technology, Cambridge, MA, USA, 1996.
- 11 Adi Shamir. IP=PSPACE. In *31st FOCS*, pages 11–15. IEEE Computer Society Press, October 1990.
- 12 Paul Valiant. Incrementally Verifiable Computation or Proofs of Knowledge Imply Time/Space Efficiency. In Ran Canetti, editor, *TCC 2008*, volume 4948 of *LNCS*, pages 1–18. Springer, Heidelberg, March 2008.
- 13 Joachim von zur Gathen and Igor E. Shparlinski. Generating safe primes. *J. Mathematical Cryptology*, 7(4):333–365, 2013. doi:10.1515/jmc-2013-5011.
- 14 Benjamin Wesolowski. Slow-timed hash functions. Cryptology ePrint Archive, Report 2018/623, 2018. URL: <https://eprint.iacr.org/2018/623>.





# Sum of Squares Lower Bounds from Symmetry and a Good Story

Aaron Potechin

University of Chicago Department of Computer Science, 5730 S. Ellis Avenue,  
John Crerar Library, Chicago, IL 60637, United States  
potechin@uchicago.edu

---

## Abstract

---

In this paper, we develop machinery which makes it much easier to prove sum of squares lower bounds when the problem is symmetric under permutations of  $[1, n]$  and the unsatisfiability of our problem comes from integrality arguments, i.e. arguments that an expression must be an integer. Roughly speaking, to prove SOS lower bounds with our machinery it is sufficient to verify that the answer to the following three questions is yes:

1. Are there natural pseudo-expectation values for the problem?
2. Are these pseudo-expectation values rational functions of the problem parameters?
3. Are there sufficiently many values of the parameters for which these pseudo-expectation values correspond to the actual expected values over a distribution of solutions which is the uniform distribution over permutations of a single solution?

We demonstrate our machinery on three problems, the knapsack problem analyzed by Grigoriev, the MOD 2 principle (which says that the complete graph  $K_n$  has no perfect matching when  $n$  is odd), and the following Turan type problem: Minimize the number of triangles in a graph  $G$  with a given edge density. For knapsack, we recover Grigoriev's lower bound exactly. For the MOD 2 principle, we tighten Grigoriev's linear degree sum of squares lower bound, making it exact. Finally, for the triangle problem, we prove a sum of squares lower bound for finding the minimum triangle density. This lower bound is completely new and gives a simple example where constant degree sum of squares methods have a constant factor error in estimating graph densities.

**2012 ACM Subject Classification** Theory of computation → Proof complexity

**Keywords and phrases** Sum of squares hierarchy, proof complexity, graph theory, lower bounds

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2019.61

**Related Version** A full version of the paper is available at <https://arxiv.org/abs/1711.11469>.

**Acknowledgements** The author would like to thank Sasha Razborov for suggesting the triangle problem and for helpful conversations. The author would also like to thank Johan Håstad, Fernando Geronimo, Annie Raymond, and anonymous reviewers for helpful comments on the paper. Finally, the author would like to thank Fernando Geronimo for helpful discussions on representation theory. This work was supported by the Simons Collaboration for Algorithms and Geometry, the NSF under agreement No. CCF-1412958, the Knut and Alice Wallenberg Foundation, the European Research Council, and the Swedish Research Council.



© Aaron Potechin;  
licensed under Creative Commons License CC-BY  
10th Innovations in Theoretical Computer Science (ITCS 2019).  
Editor: Avrim Blum; Article No. 61; pp. 61:1–61:20



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

## 1 Introduction

The sum of squares hierarchy (which we call SOS for brevity), a hierarchy of semidefinite programs first independently investigated by Shor [33], Nesterov [27], Parrilo [28], Lasserre [22], and Grigoriev [15, 16], is an exciting frontier of algorithm design, complexity theory, and proof complexity. SOS is exciting because it provides a single unified framework which can be applied to give approximation algorithms for a wide variety of combinatorial optimization problems. Moreover, SOS is conjectured to be optimal for many of these problems. In particular, SOS captures the Goemans-Williamson algorithm for MAX-CUT [13], the Goemans-Linial relaxation for sparsest cut (analyzed by Arora, Rao, and Vazirani [2]), and the subexponential time algorithm for unique games found by Arora, Barak, and Steurer [1]. More recently, SOS has been applied directly to give algorithms for several problems including planted sparse vector [5], dictionary learning [6], tensor decomposition [12, 19, 25], tensor completion [8, 29], and quantum separability [7].

That said, there are limits to the power of SOS. As shown by SOS lower bounds for constraint satisfactions problems (CSPs) [16, 32, 3, 20] and gadget reductions [34], SOS requires degree  $\Omega(n)$  (and thus exponential time) to solve most NP-hard problems. As shown by SOS lower bounds on planted clique and other planted problems [26, 10, 17, 4, 18], SOS can have difficulty distinguishing between a random input and an input which is random except for a solution which has been planted inside it. Finally, as shown by Grigoriev's SOS lower bound for the knapsack problem [15], SOS has difficulty capturing integrality arguments, i.e. arguments which say that an expression must be an integer.

In this paper, we further explore this last weakness of SOS. In particular, we develop machinery which makes it much easier to prove SOS lower bounds when the problem is symmetric and the unsatisfiability of our problem comes from integrality arguments. The usual process for proving SOS lower bounds involves finding pseudo-expectation values (see subsection 2.3) and then proving that a matrix called the moment matrix is PSD (positive semidefinite), which can be quite difficult. Roughly speaking, to prove SOS lower bounds with our machinery it is sufficient to verify that the answer to the following three questions is yes:

1. Are there natural pseudo-expectation values for the problem?
2. Are these pseudo-expectation values rational functions of the problem parameters?
3. Are there sufficiently many values of the parameters for which these pseudo-expectation values correspond to the actual expected values over a distribution of solutions which is the uniform distribution over permutations of a single solution?

We demonstrate our machinery on three problems, the knapsack problem itself, the MOD 2 principle (which says that the complete graph  $K_n$  on  $n$  vertices does not have a perfect matching when  $n$  is odd), and the following Turan-type problem: Minimize the number of triangles in a graph  $G$  with a given edge density.

### 1.1 Equations and SOS lower bounds for knapsack, the MOD 2 principle, and a triangle problem

To state our SOS lower bounds on knapsack, the MOD 2 principle, and the triangle problem, we must first express these problems as infeasible systems of polynomial equations. We do this because as we will discuss in subsection 2.3, SOS gives a proof system for proving that systems of polynomial equations over  $\mathbb{R}$  are infeasible. Our lower bounds show that SOS requires high degree to prove that the systems of equations corresponding to knapsack, the MOD 2 principle, and the triangle problem are infeasible.

For the knapsack problem, we consider the simple case when all of the weights are 1, the knapsack capacity is  $k$ , and we are asked whether it is possible to fill the knapsack to its capacity. We can express this problem with equations as follows:

1.  $\forall i, x_i^2 - x_i = 0$
2.  $\sum_{i=1}^n x_i - k = 0$ .

These equations are clearly infeasible whenever  $k \notin \mathbb{Z}$ . However, as Grigoriev [15] showed, since SOS has difficulty capturing integrality arguments, SOS requires high degree to refute these equations.

► **Theorem 1** (Grigoriev's SOS lower bound for knapsack).

*Degree  $\min\{2\lfloor \min\{k, n-k\} \rfloor + 3, n\}$  SOS fails to prove that the knapsack equations are infeasible.*

In this paper, we observe that Grigoriev's lower bound (which is tight) follows immediately from our machinery.

For the MOD 2 principle, we are asked whether the complete graph  $K_n$  has a perfect matching. To express this problem with equations, we take a variable  $x_{ij}$  for each possible edge  $(i, j)$  and we want that  $x_{ij} = 1$  if the edge  $(i, j)$  is in our matching and  $x_{ij} = 0$  otherwise. We encode this and the claim that we have a perfect matching as follows:

1. For all  $i, j \in [1, n]$  such that  $i < j$ ,  $x_{ij}^2 - x_{ij} = 0$
2. For all  $i \in [1, n]$ ,  $\sum_{j \in [1, n]: j \neq i} x_{ij} - 1 = 0$  (where we take  $x_{ij} = x_{ji}$  whenever  $i > j$ )

These equations are infeasible whenever  $n$  is odd. However, Grigoriev [16] showed that SOS requires high degree to refute these equations. While Grigoriev's lower bound is shown via a reduction from the Tseitin equations and is tight up to a constant factor, in this paper we use our machinery to obtain the following tight SOS lower bound directly.

► **Theorem 2** (SOS lower bound for the MOD 2 principle).

*Degree  $\frac{n-1}{2}$  SOS fails to prove that the equations for the MOD 2 principle are infeasible.*

For the triangle problem, we want to minimize the number of triangles in a graph with edge density  $\rho$ . For this problem, Goodman [14] showed the following lower bound.

► **Theorem 3** (Goodman's bound). *The minimal number of triangles in a graph  $G$  with  $n$  vertices and edge density  $\rho$  is at least*

$$t(n, \rho) := \binom{n}{3} - \frac{n(n-1)(1-\rho)}{6}((1+2\rho)n - 2 - 2\rho)$$

As we will discuss in the full version of this paper, this bound is tight if there is an integer  $k$  such that

1.  $\frac{n}{k} - 1 = (1 - \rho)(n - 1)$
2.  $n$  is divisible by  $k$ .

If so, then we can take  $G$  to have  $k$  independent sets of size  $\frac{n}{k}$  and have all of the edges between different independent sets, which minimizes the number of triangles in  $G$  and matches Goodman's bound. Otherwise, Goodman's bound cannot be achieved.

To express this problem using equations, we again create a variable  $x_{ij}$  for each possible edge  $(i, j)$  and we want  $x_{ij} = 1$  if the edge  $(i, j)$  is in the graph and  $x_{ij} = 0$  if the edge  $(i, j)$  is not in the graph. We encode this, the requirement the edge density is  $\rho$ , and the claim that Goodman's bound can be achieved with the following equations

1. For all  $i, j \in [1, n]$  such that  $i < j$ ,  $x_{ij}^2 - x_{ij} = 0$
2.  $\sum_{i, j \in [1, n]: i < j} x_{ij} - \rho \binom{n}{2} = 0$
3.  $\sum_{i, j, k \in [1, n]: i < j < k} x_{ij}x_{ik}x_{jk} - t(n, \rho) = 0$  where  $t(n, \rho) = \binom{n}{3} - \frac{n(n-1)(1-\rho)}{6}((1+2\rho)n - 2 - 2\rho)$

Using our machinery, we show the following SOS lower bound which is completely new and was the motivation for developing our machinery.

► **Theorem 4** (SOS lower bound for the triangle problem).

Letting  $k$  be the number such that  $\frac{n}{k} - 1 = (1 - \rho)(n - 1)$ , degree  $\lfloor \min\{k, \frac{n}{k}\} \rfloor + 1$  SOS fails to refute the triangle problem equations.

## 1.2 Relation to previous work on symmetry and SOS

There is a considerable body of prior research on symmetry and SOS. Several works built on the difficulty on knapsack and/or further investigated symmetric polynomials on the variables  $\{x_1, \dots, x_n\}$ . Laurent [23] used the difficulty of knapsack to show that degree  $\lfloor \frac{n}{2} \rfloor$  SOS is required to capture the CUT polytope of the complete graph. Bleckherman, Gouveia, and Pfeiffer [9] used the difficulty of knapsack to construct degree 4 polynomials which are non-negative but cannot be written as a sum of squares of low degree *rational* functions. Lee, Prakash, Wolf, and Yuen [24] showed that there are symmetric non-negative polynomials on the variables  $\{x_1, \dots, x_n\}$  which cannot be approximated with low degree sums of squares. Kurpisz, Leppänen, and Mastroiilli [21] gave a general criterion for determining if a symmetric polynomial on  $\{x_1, \dots, x_n\}$  is a sum of squares or not.

While these prior works give more precise results for symmetric problems on the variables  $\{x_1, \dots, x_n\}$ , they do not show how to handle problems which are symmetric under permutations of  $[1, n]$  but have variables such as  $\{x_{ij} : i < j\}$  which depend on 2 or more indices. Thus, these prior works are incomparable with this work.

Another line of research on symmetry and SOS which is more closely connected to this work uses symmetry to reduce the algorithmic complexity of implementing SOS. Gatermann and Parrilo [11] showed how representation theory can be used to greatly reduce the search space for pseudo-expectation values, allowing SOS to be run more efficiently on symmetric problems. Recently, Raymond et. al. [30] combined the analysis of Gatermann and Parrilo with Razborov’s flag algebras [31] to show that in the case of  $k$ -subset hypercubes, the resulting semidefinite program has size which is independent of  $n$ . These results are quite general and apply to all of the problems we are considering. That said, these results do not tell us how to find or verify pseudo-expectation values by hand, which is generally what is needed for SOS lower bounds.

In this paper, we show how the representation theory which allows Gatermann and Parrilo [11] and Raymond et. al. [30] to dramatically reduce the size of the semidefinite programs for SOS on symmetric problems can also be used to help prove theoretical SOS lower bounds on symmetric problems. In particular, Theorem 21, which is a crucial part of our machinery, essentially follows from Corollary 2.6 of Raymond et. al. [30]. We obtain our lower bounds by combining this theorem with the additional assumption that the unsatisfiability of the problem we are analyzing comes from integrality arguments.

## 1.3 Paper outline

The remainder of the paper is organized as follows. In Section 2, we give some preliminaries. In Section 3 we describe how we can find candidate pseudo-expectation values from stories. In Section 4 we highlight how symmetry is useful for proving SOS lower bounds even without additional assumptions. In Section 5, we rigorously define what stories and good stories are and show that good stories imply SOS lower bounds. Finally, in Section 6, we show a method for verifying that stories are good stories.

## 2 Preliminaries

Before we can describe our machinery, we must first give some preliminaries. We begin by describing the class of symmetric problems which our machinery can be applied to. We then define the sum of squares hierarchy and discuss some notation for the paper.

### 2.1 Symmetric problems

► **Definition 5.** We make the following assumptions about the problem  $P$  we are analyzing:

1. We assume that  $P$  is a problem about hypergraphs  $G$  with vertices  $V(G) = [1, n]$  and a set of possible hyperedges  $E_P$ . We view the hyperedges  $e \in E_P$  as subsets of  $[1, n]$  which may be unordered or ordered depending on  $P$ . If all of these subsets have the same size  $t \geq 1$  then we say that the problem  $P$  has arity  $t$ .
2. We assume that  $P$  has variables  $\{x_e : e \in E_P\}$  and  $P$  is a YES/NO question which is described by a set of problem equations  $\{s_i(\{x_e : e \in E_P\}) = 0\}$ . The answer to  $P$  is YES if all of these equations can be satisfied simultaneously and NO otherwise.
3. We assume that the set  $E_P$  of possible hyperedges and the set  $\{s_i(\{x_e : e \in E_P\}) = 0\}$  of problem equations are both symmetric under permutations of  $[1, n]$ .

If a problem  $P$  satisfies all of these assumptions then we say that  $P$  is a symmetric hypergraph problem. Since we only consider problems of this type, for brevity we will just say symmetric problem rather than symmetric hypergraph problem.

► **Example 6.** Symmetric problems  $P$  of arity 1 are YES/NO questions on the variables  $\{x_1, \dots, x_n\}$  which are symmetric under permutations of  $[1, n]$ .

► **Example 7.** For symmetric problems  $P$  of arity 2,  $E_P$  is the set of subsets of  $[1, n]$  of size 2. If the subsets in  $E_P$  are unordered then  $G$  is an undirected graph and we have variables  $\{x_{ij} : i, j \in [1, n], i \neq j\}$  where we take  $x_{ji} = x_{ij}$ . If the subsets in  $E_P$  are ordered then  $G$  is a directed graph and we have distinct variables  $\{x_{ij} : i, j \in [1, n], i \neq j\}$ .

► **Remark.** While our machinery can handle symmetric problems of any arity, the examples we focus on all have arity 1 or 2. Knapsack with unit weights has arity 1 while the MOD 2 principle and the triangle problem have arity 2 and are about undirected graphs.

► **Remark.** Since our machinery is based on polynomial interpolation, it is important that the symmetric problem  $P$  does not have inequalities as well as equalities. If  $P$  has inequalities then our machinery does not immediately give an SOS lower bound and more analysis is needed.

### 2.2 Index degree

For our results, rather than considering the degree of a polynomial  $f$ , it is more natural to consider the largest number of indices mentioned in any one monomial of  $f$ . We call this the index degree of  $f$ .

► **Definition 8 (Index degree).**

1. Given a monomial  $p = \prod_{e \in E_P} x_e$ , we define  $I(p) = \{i : \exists e \in E_P : i \in e\}$  and we define the index degree of  $p$  to be

$$\text{indexdeg}(p) = \text{indexdeg}_{[1, n]}(p) = |I(p)|$$

In other words,  $\text{indexdeg}(p)$  is the number of indices which  $p$  depends on.

2. Given a polynomial  $f$ , if  $f = \sum_j c_j p_j$  is the decomposition of  $f$  into monomials then we define the index degree of  $f$  to be  $\text{indexdeg}(f) = \max_j \{\text{indexdeg}(p_j)\}$

► **Example 9.** If  $p$  is the monomial  $p = x_{12}x_{34}$  then  $p$  has degree 2 and index degree 4.

► **Example 10.** If  $f = x_{12}x_{13} + x_{24}^4$  then  $f$  has degree 4 and index degree 3.

We will also need an analagous definition where we only consider the indices outside of a subset  $I \subseteq [1, n]$ .

► **Definition 11** (Index degree outside of  $I$ ). Let  $I \subseteq [1, n]$  be a subset of indices.

1. Given a monomial  $p = \prod_{e \in E_p} x_e$ , we define the index degree of  $p$  on  $[1, n] \setminus I$  to be

$$\text{indexdeg}_{[1, n] \setminus I}(p) = |I(p) \setminus I|$$

In other words,  $\text{indexdeg}_{[1, n] \setminus I}(p)$  is the number of indices in  $[1, n] \setminus I$  which  $p$  depends on.

2. Given a polynomial  $f$ , if  $f = \sum_j c_j p_j$  is the decomposition of  $f$  into monomials then we define the index degree of  $f$  on  $[1, n] \setminus I$  to be  $\text{indexdeg}_{[1, n] \setminus I}(f) = \max_j \{\text{indexdeg}_{[1, n] \setminus I}(p_j)\}$

### 2.3 SOS and pseudo-expectation values

We now define SOS and pseudo-expectation values, which are used to prove SOS lower bounds. One way to describe SOS is through SOS/Positivstellensatz proofs, which are defined as follows:

► **Definition 12.** Given a system of polynomial equations  $\{s_i = 0\}$  over  $\mathbb{R}$ , an index degree  $d$  SOS/Positivstellensatz proof of infeasibility is an equality of the form

$$-1 = \sum_i f_i s_i + \sum_j g_j^2$$

where

1.  $\forall i, \text{indexdeg}(f_i) + \text{indexdeg}(s_i) \leq d$
2.  $\forall j, \text{indexdeg}(g_j) \leq \frac{d}{2}$

► **Remark.** This is a proof of infeasibility because the terms  $f_i s_i$  should all be 0 by the problem equations and the terms  $g_j^2$  must all be non-negative, so they can't possibly sum to  $-1$  if all of the problem equations are satisfied.

► **Definition 13.** Index degree  $d$  SOS gives the following feasibility test for whether a system of polynomial equations over  $\mathbb{R}$  is feasible or not. If there is an index degree  $d$  Positivstellensatz proof of infeasibility then index degree  $d$  SOS says NO. Otherwise, index degree  $d$  SOS says YES.

► **Remark.** Index degree  $d$  SOS may give false positives by failing to say NO on systems of equations which are infeasible but will never give a false negative.

In this paper, we show SOS lower bounds for the infeasible systems of equations described in subsection 2.1 by showing that for small  $d$  there is no index degree  $d$  Positivstellensatz proof of infeasibility for our system of equations. This can be done with index degree  $d$  pseudo-expectation values, which are defined as follows:

► **Definition 14.** Given a system of polynomial equations  $\{s_i = 0\}$  over  $\mathbb{R}$ , index degree  $d$  pseudo-expectation values are a linear mapping  $\tilde{E}$  from polynomials of index degree  $\leq d$  to  $\mathbb{R}$  which satisfies the following conditions:

1.  $\tilde{E}[1] = 1$
2.  $\forall i, f, \tilde{E}[f s_i] = 0$  whenever  $\text{indexdeg}(f) + \text{indexdeg}(s_i) \leq d$
3.  $\forall g, \tilde{E}[g^2] \geq 0$  whenever  $\text{indexdeg}(g) \leq \frac{d}{2}$

► **Proposition 15.** *If there are index degree  $d$  pseudo-expectation values  $\tilde{E}$  for a system of polynomial equations  $s_1 = 0, s_2 = 0, \dots$  over  $\mathbb{R}$ , then there is no index degree  $d$  Positivstellensatz proof of infeasibility for these equations.*

**Proof.** Assume that we have both index degree  $d$  pseudo-expectation values and an index degree  $d$  Positivstellensatz proof of infeasibility. Applying the pseudo-expectation values to the Positivstellensatz proof, we get the following contradiction:

$$-1 = \tilde{E}[-1] = \sum_i \tilde{E}[f_i s_i] + \sum_j \tilde{E}[g_j^2] \geq 0 \quad \blacktriangleleft$$

► **Remark.** Condition 3 of definition 14 is equivalent to the statement that the moment matrix  $M$  is PSD (positive semidefinite) where  $M$  is indexed by monomials  $p, q$  of index degree  $\leq \frac{d}{2}$  and has entries  $M_{pq} = \tilde{E}[pq]$ . Proving SOS lower bounds usually involves proving that  $M \succeq 0$ , which can be quite difficult. In this paper we can instead analyze  $\tilde{E}[g^2]$  more directly.

► **Remark.** The idea behind pseudo-expectation values is that they should mimic actual expected values over a distribution of solutions. In particular, as shown by the following proposition, if  $\tilde{E}$  comes from a distribution over actual solutions then it automatically gives pseudo-expectation values. This fact is crucial for our results.

► **Proposition 16.** *If the equations  $\{s_i = 0\}$  are feasible over  $\mathbb{R}$  and  $\Omega$  is a probability distribution over actual solutions then the linear mapping  $\tilde{E}[p] = E_\Omega[p]$  gives index degree  $d$  pseudo-expectation values for these equations for all  $d$ .*

**Proof.** Observe that:

1. For any  $x \sim \Omega$ ,  $1 = 1$ . Thus,  $\tilde{E}[1] = E_\Omega[1] = 1$ .
2. For any  $x \sim \Omega$ , for all  $i, f$ ,  $f(x)s_i(x) = 0$ . Thus, for all  $i, f$ ,  $\tilde{E}[f s_i] = E_\Omega[f s_i] = 0$ .
3. For any  $x \sim \Omega$ , for all  $g$ ,  $g(x)^2 \geq 0$ . Thus, for all  $g$ ,  $\tilde{E}[g^2] = E_\Omega[g^2] \geq 0$ . ◀

## 2.4 Sequences of distinct indices

We will need the following definitions about sequences of distinct indices in  $[1, n]$ .

► **Definition 17** (Operations on sequences).

1. Given a sequence of distinct indices  $A = (i_1, \dots, i_m)$ , we define the set  $I_A$  to be  $I_A = \{i_1, \dots, i_m\}$ . In other words,  $I_A$  is just  $A$  without the ordering.
2. Given two sequences of distinct indices  $A = (i_1, \dots, i_{m_1})$  and  $B = (i'_1, \dots, i'_{m_2})$ , we say that  $A \subseteq B$  if  $m_1 \leq m_2$  and  $\forall j \in [1, m_1], i'_j = i_j$ .
3. Given two sequences of distinct indices  $A = (i_1, \dots, i_{m_1})$  and  $B = (i'_1, \dots, i'_{m_2})$  such that  $I_A \cap I_B = \emptyset$ , we define  $A \cup B$  to be the sequence  $A \cup B = (i_1, \dots, i_{m_1}, i'_1, \dots, i'_{m_2})$

In this paper, we will never consider sequences of indices which are not distinct, so we assume without stating it explicitly that all of our sequences contain distinct indices.

## 3 Finding pseudo-expectation values: Stories and a verifier/adversary game for SOS

In this section, we describe a verifier/adversary game which we use to find pseudo-expectation values and deduce SOS lower bounds. We then describe how the adversary can play this game using stories and describe the resulting pseudo-expectation values for knapsack, the MOD 2 principle, and the triangle problem.



## 61:8 Sum of Squares Lower Bounds from Symmetry and a Good Story

The verifier/adversary game is as follows. The verifier queries sequences of indices  $\{A_i\}$ . For each sequence of indices  $A = (i_1, \dots, i_m)$  the verifier queries, for each  $j \in [1, m]$  and every possibility for what happens with the previous indices  $(i_1, \dots, i_{j-1})$ , the adversary must provide a probability distribution for what happens with the index  $i_j$ . Taken together, these answers give a probability distribution for all of the possibilities for what happens with the indices in  $A$ . From these probability distributions, we can obtain pseudo-expectation values.

The verifier wins if he/she detects one of the following flaws in the adversary's answers

1. The adversary gives a probability for some event which is either negative or undefined.
2. The adversary's answers do not result in well-defined pseudo-expectation values because they are inconsistent. More precisely, there exist two sequences of indices  $A = (i_1, \dots, i_m)$  and  $A' = (i'_1, \dots, i'_m)$  such that  $A'$  and  $A$  are equal as sets (i.e.  $\{i'_1, \dots, i'_m\}$  is a permutation of  $\{i_1, \dots, i_m\}$ ) and the resulting probability distributions for what happens with the indices  $\{i_1, \dots, i_m\}$  do not match.
3. The adversary's answers result in pseudo-expectation values such that some problem equation  $s_i = 0$  is violated i.e.  $\tilde{E}[fs_i] \neq 0$  for some polynomial  $f$ .

If the verifier is unable to find such a flaw then the adversary wins.

► **Remark.** Roughly speaking, when we say that the adversary specifies what happens with a set of indices  $I$  we mean that the adversary assigns values to all variables  $x_e$  such that the indices of  $e$  are contained in  $I$ . We make this more precise in Section 5.

The adversary often has a strategy for this game based on a story for what happens with the indices. For the problems we are analyzing, the adversary's stories are as follows:

1. Knapsack: We set  $k$  out of the  $n$   $x_i$  to be 1 and set the rest to 0.
2. The MOD 2 principle: For each vertex  $i$ , the perfect matching contains precisely one of the edges which are incident to  $i$ .
3. The triangle problem: We have  $k$  independent sets of size  $\frac{n}{k}$ .

► **Remark.** The adversary's stories are not convincing to us, as we can understand integrality arguments. However, the adversary just has to fool SOS, which is poor at capturing integrality arguments.

We now demonstrate how these stories naturally give probability distributions for what happens with the indices and thus give pseudo-expectation values.

► **Example 18 (Knapsack).** For knapsack, if the verifier first queries vertex  $i$ , the adversary says that  $x_i = 1$  with probability  $\frac{k}{n}$  and  $x_i = 0$  with probability  $\frac{n-k}{n}$ . Thus, according to the adversary the expected value of  $x_i$  is  $\frac{k}{n}$  so we take  $\tilde{E}[x_i] = \frac{k}{n}$

If the verifier then queries  $x_j$ , if we have  $x_i = 1$  then the adversary says that  $x_j = 1$  with probability  $\frac{k-1}{n-1}$  and  $x_j = 0$  with probability  $\frac{n-k}{n-1}$  as the adversary wants to set  $k-1$  of the remaining  $n-1$  variables to 1. If we have  $x_i = 0$  then the adversary instead says that  $x_j = 1$  with probability  $\frac{k}{n-1}$  and  $x_j = 0$  with probability  $\frac{n-k-1}{n-1}$  as the adversary wants to set  $k$  of the remaining  $n-1$  variables to 1. Thus, according to the adversary the expected value of  $x_i x_j$  is  $\frac{k(k-1)}{n(n-1)}$  so we take  $\tilde{E}[x_i x_j] = \frac{k(k-1)}{n(n-1)}$ .

Following similar logic, for all  $I \subseteq [1, n]$  such that  $|I| \leq d$ ,  $\tilde{E}[\prod_{i \in I} x_i] = \frac{\binom{k}{|I|}}{\binom{n}{|I|}}$

► **Example 19 (MOD 2 principle).** For the MOD 2 principle, if the verifier first queries  $i$ , the adversary gives no information because there is nothing distinguishing  $i$  from other vertices. If the verifier then queries  $j$ , the adversary says that  $x_{ij} = 1$  with probability  $\frac{1}{n-1}$  and  $x_{ij} = 0$  with probability  $\frac{n-2}{n-1}$  because the adversary wants to match 1 out of the remaining  $n-1$  vertices with  $i$ . Thus, we take  $\tilde{E}[x_{ij}] = \frac{1}{n-1}$

We now consider  $\tilde{E}[x_{ij}x_{kl}]$  where  $i, j, k, l$  are all distinct.  $x_{ij}x_{kl} = 0$  unless  $x_{ij} = 1$  so we can focus on the case when  $x_{ij} = 1$ , which according to the adversary happens with probability  $\frac{1}{n-1}$ . In this case, if the verifier queries  $k$ , the adversary gives no additional information because there is nothing distinguishing  $k$  from other vertices in  $[1, n] \setminus (i, j)$ . If the verifier then queries  $l$ , the adversary says that  $x_{kl} = 1$  with probability  $\frac{1}{n-3}$  and  $x_{kl} = 0$  with probability  $\frac{n-4}{n-3}$  because the adversary wants to match 1 of the  $n-3$  remaining vertices with  $k$ . Thus, we take  $\tilde{E}[x_{ij}x_{kl}] = \frac{1}{(n-1)(n-3)}$

Following similar logic, we obtain that for all  $E \subseteq \{(i, j) : i, j \in [1, n], i < j\}$  such that  $|E| \leq d$ ,  $\tilde{E}[\prod_{(i,j) \in E} x_{ij}] = \frac{1}{\prod_{j=1}^{|E|} (n-2j+1)}$  if  $E$  is a partial matching and  $\tilde{E}[\prod_{(i,j) \in E} x_{ij}] = 0$  otherwise.

► **Example 20 (Triangle Problem).** For the triangle problem, if the verifier first queries  $i$ , the adversary gives no information because there is nothing distinguishing  $i$  from other vertices. If the verifier then queries  $j$ , the adversary says that  $j$  is in the same independent set as  $i$  with probability  $\frac{\frac{n}{k}-1}{n-1}$  and is in a different independent set with probability  $\frac{n-\frac{n}{k}}{n-1}$ .

If the verifier then queries  $k$ , if  $i, j$  are in the same independent set then the adversary says that  $k$  is in the same independent set as  $i, j$  with probability  $\frac{\frac{n}{k}-2}{n-2}$  and is in a different independent set with probability  $\frac{n-\frac{n}{k}}{n-2}$ . If  $i, j$  are in different independent sets then the adversary says that  $k$  is in the same independent set as  $i$  with probability  $\frac{\frac{n}{k}-1}{n-2}$ ,  $k$  is in the same independent set as  $j$  with probability  $\frac{\frac{n}{k}-1}{n-2}$ , and  $k$  is in a different independent set with probability  $\frac{n-2\frac{n}{k}}{n-2}$ . Thus, the adversary gives the following probabilities for what happens with  $i, j, k$ :

1. The probability that  $i, j, k$  are all in the same independent set is  $\frac{(\frac{n}{k}-1)(\frac{n}{k}-2)}{(n-1)(n-2)}$
2. The probability that  $i, j$  are in the same independent set and  $k$  is in a different independent set is  $\frac{(\frac{n}{k}-1)(n-\frac{n}{k})}{(n-1)(n-2)}$ . This is also the probability that  $i, k$  are in the same independent set and  $j$  is in a different independent set and the probability that  $j, k$  are in the same independent set and  $i$  is in a different independent set.
3. The probability that  $i, j, k$  are all in different independent sets is  $\frac{(n-\frac{n}{k})(n-2\frac{n}{k})}{(n-1)(n-2)}$

This gives the following pseudo-expectation values:

1.  $\tilde{E}[x_{ij}] = \frac{n-\frac{n}{k}}{n-1}$
2.  $\tilde{E}[x_{ij}x_{ik}] = \tilde{E}[x_{ij}x_{jk}] = \tilde{E}[x_{ik}x_{jk}] = \frac{(\frac{n}{k}-1)(n-\frac{n}{k})}{(n-1)(n-2)} + \frac{(n-\frac{n}{k})(n-2\frac{n}{k})}{(n-1)(n-2)} = \frac{(n-\frac{n}{k})(n-\frac{n}{k}-1)}{(n-1)(n-2)}$
3.  $\tilde{E}[x_{ij}x_{ik}x_{jk}] = \frac{(n-\frac{n}{k})(n-2\frac{n}{k})}{(n-1)(n-2)}$

► **Remark.** For the triangle problem, it is difficult to write down the general expression for  $\tilde{E}$  explicitly. Fortunately, as we will show, we can verify the conditions of Definition 14 based on the story for  $\tilde{E}$

## 4 Symmetry and SOS lower bounds

In this section, we highlight how symmetry can help prove SOS lower bounds even without additional assumptions. In particular, we have the following theorem which essentially follows from Corollary 2.6 of [30].

► **Theorem 21.** *If  $\tilde{E}$  is a linear map from polynomials to  $\mathbb{R}$  which is symmetric with respect to permutations of  $[1, n]$  then for any polynomial  $g$ , we can write*

$$\tilde{E}[g^2] = \sum_{I \subseteq [1, n], j: |I| \leq \text{indexdeg}(g)} \tilde{E}[g_{Ij}^2]$$

where for all  $I, j$ ,

## 61:10 Sum of Squares Lower Bounds from Symmetry and a Good Story

1.  $g_{I_j}$  is symmetric with respect to permutations of  $[1, n] \setminus I$ .
2.  $\text{indexdeg}(g_{I_j}) \leq \text{indexdeg}(g)$
3.  $\forall i \in I, \sum_{\sigma \in S_{[1, n] \setminus (I \setminus \{i\})}} \sigma(g_{I_j}) = 0$

Theorem 21 is very useful for proving SOS lower bounds on symmetric problems because it implies that instead of checking that  $\tilde{E}[g^2] \geq 0$  for all polynomials of index degree  $\leq \frac{d}{2}$ , it is sufficient to check polynomials which are symmetric under permutations of all but  $\frac{d}{2}$  indices. However, despite its simplicity, Theorem 21 is quite deep. To prove Theorem 21, we must carefully decompose  $g$  and then use symmetry to analyze all of the non-square terms of  $g^2$  and either eliminate them or reduce them to square terms. Fortunately, this has already been done by Corollary 2.6 of [30] using representation theory. We now sketch how Theorem 21 follows from Corollary 2.6 of [30].

### Proof sketch of Theorem 21 using Corollary 2.6 of [30].

► **Definition 22** (Definition 2.1 of [30]). If  $\oplus_{\lambda} V_{\lambda}$  is the isotypic decomposition of the vector space of polynomials of degree  $\leq d$  and  $\tau_{\lambda}$  is a tableau of shape  $\lambda$ , define

$$W_{\tau_{\lambda}} := V_{\lambda}^{\mathcal{R}_{\tau_{\lambda}}}$$

to be the subspace of the isotypic  $V_{\lambda}$  fixed by the action of the row group  $\mathcal{R}_{\tau_{\lambda}}$  (which keeps each row of  $\tau_{\lambda}$  fixed but may permute the elements within each row of  $\tau_{\lambda}$ )

Corollary 2.6 of [30] (rephrased slightly) says the following:

► **Corollary 23** (Corollary 2.6 of [30]). *Suppose  $p$  is a polynomial on the variables  $\{x_{ij} : i, j \in [1, n], i < j\}$  such that  $p$  is symmetric under permutations of  $[1, n]$  and  $p$  can be written as a sum of squares of polynomials of degree  $\leq d$ . For each partition  $\lambda \vdash n$ , fix a tableau  $\tau_{\lambda}$  of shape  $\lambda$  and choose a vector space basis  $\{b_1^{\tau_{\lambda}}, \dots, b_{m_{\lambda}}^{\tau_{\lambda}}\}$  for  $W_{\tau_{\lambda}}$ . Then for each partition  $\lambda \in \Lambda$ , there exists an  $m_{\lambda} \times m_{\lambda}$  PSD matrix  $Q_{\lambda}$  such that*

$$p = \sum_{\lambda \in \Lambda} \text{tr}(Q_{\lambda} Y^{\tau_{\lambda}})$$

where  $\Lambda := \{\lambda \vdash n : \lambda \geq_{lex} (n - 2d, 1^{2d})\}$  and  $Y_{ij}^{\tau_{\lambda}} := \text{sym}(b_i^{\tau_{\lambda}} b_j^{\tau_{\lambda}})$

Using Corollary 2.6 of [30], we can prove Theorem 21 as follows. Since  $\tilde{E}$  is symmetric,  $\tilde{E}[g^2] = \tilde{E}[\text{sym}(g^2)]$  where  $\text{sym}(g^2) = \frac{1}{n!} \sum_{\sigma \in S_n} (\sigma(g))^2$ . Since  $\text{sym}(g^2)$  is symmetric and a sum of squares, by Corollary 2.6 of [30], there exist PSD matrices  $Q_{\lambda}$  such that

$$\tilde{E}[g^2] = \sum_{\lambda \in \Lambda} \tilde{E}[\text{tr}(Q_{\lambda} Y^{\tau_{\lambda}})]$$

Since  $\tilde{E}$  is symmetric, this implies that

$$\tilde{E}[g^2] = \sum_{\lambda \in \Lambda} \tilde{E}[\text{tr}(Q_{\lambda} Y'^{\tau_{\lambda}})]$$

where  $Y'^{\tau_{\lambda}} := b_i^{\tau_{\lambda}} b_j^{\tau_{\lambda}}$ . Now consider each  $\lambda \in \Lambda$  separately and observe that since  $Q_{\lambda} \geq 0$ , we can write  $Q_{\lambda} = \sum_j q^j q^{jT}$  for some vectors  $\{q^1, \dots, q^{m_{\lambda}}\}$ . Thus,

$$\text{tr}(Q_{\lambda} Y^{\tau_{\lambda}}) = \text{tr}\left(\sum_j q^j q^{jT} b^{\tau_{\lambda}} b^{\tau_{\lambda}T}\right) = \sum_j q^{jT} b^{\tau_{\lambda}} b^{\tau_{\lambda}T} q^j = \sum_j \left(\sum_{i \in m_{\lambda}} q_i^j b_i^{\tau_{\lambda}}\right)^2$$

which means we can reexpress  $\text{sym}(g^2)$  as a sum of squares, each of which has the form  $(\sum_{i=1}^{m_{\lambda}} c_i b_i^{\tau_{\lambda}})^2$  for some partition  $\lambda \vdash n$ , tableau  $\tau_{\lambda}$  of shape  $\lambda$ , and coefficients  $\{c_i\}$

For each square  $(\sum_{i=1}^{m_\lambda} c_i b_i^{\tau_\lambda})^2$ , let  $I$  be the set of indices which are not in the top row of  $\tau_\lambda$ . To show the first statement of Theorem 21, observe that permuting the indices of  $[1, n] \setminus I$  is just permuting the top row of  $\tau_\lambda$ . By definition, the elements of  $W_{\tau_\lambda}$  are all invariant under such permutations, so  $\sum_{i=1}^{m_\lambda} c_i b_i^{\tau_\lambda}$  is invariant under permutations of  $[1, n] \setminus I$ , as needed.

► **Remark.** In the setting of Corollary 2.6 of [30] the variables are  $\{x_{ij} : i, j \in [1, n], i < j\}$  so if  $g$  has degree  $d$ ,  $g$  can have index degree  $2d$  which matches the fact that  $\Lambda := \{\lambda \vdash n : \lambda \geq_{lex} (n - 2d, 1^{2d})\}$ . To prove Theorem 21 as stated using Corollary 2.6 of [30], Corollary 2.6 of [30] must be restated in terms of index degree and the proof adjusted accordingly.

The second statement of Theorem 21 is trivial as all of the  $b_i^{\tau_\lambda}$  are in the vector space of polynomials of degree  $\leq d$  and thus index degree  $\leq 2d$ .

To show the third statement of Theorem 21, we need to prove the following lemma

► **Lemma 24.** *For any  $\tau_\lambda$ , letting  $I$  be the set of indices which are not in the top row of  $\tau_\lambda$ , for any  $i \in I$  and any  $p \in W_{\tau_\lambda}$ ,*

$$\sum_{\sigma \in S_{([1, n] \setminus I) \cup \{i\}}} \sigma(p) = 0$$

**Proof sketch.** This lemma follows from the following claim:

► **Definition 25.** Define  $U_r = \text{span}\{p : \exists I \subseteq [1, n] : |I| = r, \forall \sigma \in S_{[1, n] \setminus I}, \sigma(p) = p\}$  and define

$$V_r = U_r / U_{r-1} = \text{span}\{p : \exists I \subseteq [1, n] : |I| = r, \forall \sigma \in S_{[1, n] \setminus I}, \sigma(p) = p, \\ \forall J \subseteq [1, n] : |J| \leq r - 1, \sum_{\sigma \in S_{[1, n] \setminus J}} \sigma(p) = 0\}$$

► **Claim 26.**  $V_r = \oplus_{\lambda: \text{The top row of } \lambda \text{ has length } n-r} V_\lambda$

Assuming this claim, for any  $\tau_\lambda$ , letting  $I$  be the set of indices which are not in the top row of  $\tau_\lambda$ , for any  $p \in W_{\tau_\lambda} \subseteq V_\lambda$  and any  $J$  such that  $|J| < |I|$ ,  $\sum_{\sigma \in S_{[1, n] \setminus J}} \sigma(p) = 0$ . Taking  $J = I \setminus \{i\}$ , the result follows.

We defer the proof of this claim to the full version. ◀

◀

However, Corollary 2.6 of [30] does not give us an explicit expression for  $\tilde{E}[g^2]$ , so we can ask whether we can obtain an explicit expression for  $\tilde{E}[g^2]$ . It turns out that there is such an expression but it is quite complicated. For an alternative proof of Theorem 21 which is explicit and combinatorial but technical, see the full version of this paper.

## 5 Sum of squares lower bounds from symmetry and a good story

In this section, we show how strategies for the verifier/adversary game described in section 3 with certain properties, which we call good stories, imply SOS lower bounds.

## 5.1 Stories

In this subsection, we rigorously define what we mean by stories. Once the definition is understood, stories are generally recognizable on sight.

► **Definition 27.** Given a subset  $I$  of  $[1, n]$ , we define  $\mathcal{P}_I$  to be the set of all polynomials which only depend on the variables  $\{x_e : e \subseteq I\}$

► **Definition 28 (Stories).** Let  $P$  be the problem we are analyzing and let  $A = (i_1, \dots, i_m)$  be a sequence of indices. We say that a strategy  $S$  for adversary is a level  $n'$  story for  $(P, A)$ , describing what will happen with the remaining indices after we have already queried  $A$ , if the following is true:

1.  $n' \leq n - |I_A|$
  2.  $S$  specifies what happened with the indices in  $A$ . More precisely, there is a linear map  $\tilde{E}_{S,A} : \mathcal{P}_{I_A} \rightarrow \mathbb{R}$  corresponding to  $S$
  3. For all  $i \in [1, n] \setminus I_A$ ,  $S$  gives values  $\{p_{ij}\}$  for the probabilities of level  $n' - 1$  stories  $S_{ij}$  for  $(P, A \cup (i))$ .
  4. We have that for all  $i \in [1, n] \setminus I_A$ ,  $\sum_j p_{ij} = 1$  and  $\forall f \in \mathcal{P}_{I_A}, \forall j, \tilde{E}_{S,A}[f] = \tilde{E}_{S_{ij},(A \cup (i))}[f]$
- Given a level  $n'$  story  $S$  for  $(P, A)$ , for all sequences  $B$  such that  $A \subseteq B$ , letting  $i$  be the next element in  $B$  after  $A$ , we define  $\tilde{E}_{S,B} = \sum_j p_{ij} \tilde{E}_{S_{ij},B}$

► **Remark.** Note that we do not require the values  $p_{ij}$  to be non-negative in this definition.

► **Remark.** For all of our examples we will have that  $n' = n - |I_A|$  but we do not force this to be the case in the definition.

## 5.2 Useful story properties part 1

We now define several properties our stories may have which are useful for proving SOS lower bounds. In Section 6 we will describe a method for verifying these properties.

The first property we want is that our story  $S$  gives the same linear map  $\tilde{E}_S$  regardless of the order we query the indices.

► **Definition 29.** We say that a level  $n'$  story  $S$  for  $(P, A)$  is self-consistent if whenever  $B, B'$  are sequences such that  $A \subseteq B, A \subseteq B', |I_B \setminus I_A| \leq n', |I_{B'} \setminus I_A| \leq n'$ ,

$$\forall p \in \mathcal{P}_{I_B \cap I_{B'}}, \tilde{E}_{S,B}[p] = \tilde{E}_{S,B'}[p]$$

If  $S$  is self-consistent then we define  $\tilde{E}_S : \{f : \text{indexdeg}_{[1,n] \setminus I_A}(f) \leq n'\} \rightarrow \mathbb{R}$  to be the linear map such that for all monomials  $p$  such that  $\text{indexdeg}_{[1,n] \setminus I_A}(p) \leq n'$ , for any sequence  $B$  of length at most  $n'$  such that  $I_B \cap I_A = \emptyset$  and  $B$  contains all indices in variables of  $p$  which are not in  $I_A$ ,  $\tilde{E}_S[p] = \tilde{E}_{S,(A \cup B)}[p]$

A second property we want is that our story sounds like we are taking the expected values over the uniform distribution of permutations of a single input graph  $G_0$ . To make this precise, we note a useful property such expected values have. We then define single-graph mimics to be stories/pseudo-expectation values which also have this property.

► **Proposition 30.** If  $\Omega$  is the trivial distribution consisting of a single graph  $G_0$  then for any polynomials  $f$  and  $g$ ,  $E_\Omega[fg] = E_\Omega[f]E_\Omega[g]$

**Proof.**  $E_\Omega[fg] = f(G_0)g(G_0) = E_\Omega[f]E_\Omega[g]$  ◀

► **Proposition 31.** If  $\Omega$  is the uniform distribution over all permutations of a single graph  $G_0$  then for all symmetric polynomials  $f$  and  $g$ ,  $E_\Omega[fg] = E_\Omega[f]E_\Omega[g]$ .

**Proof.** For any symmetric polynomial  $h$  and any permutation  $\sigma$ ,  $h(\sigma(G_0)) = h(G_0)$  which implies that  $E_\Omega[h] = h(G_0)$ . Thus, we again have that  $E_\Omega[fg] = f(G_0)g(G_0) = E_\Omega[f]E_\Omega[g]$ , as needed.  $\blacktriangleleft$

► **Remark.** The property that  $E[fg] = E[f]E[g]$  for all symmetric polynomials  $f, g$  is useful because it immediately implies that for all symmetric polynomials  $g$ ,  $E[g^2] = (E[g])^2 \geq 0$ .

We now define single graph mimics.

► **Definition 32.** Let  $P$  be a symmetric problem with equations  $\{s_i = 0\}$  and let  $I$  be a subset of  $[1, n]$ . We say that  $\tilde{E}$  is a level  $n'$  single graph mimic for  $P$  on  $[1, n] \setminus I$  if the following conditions hold:

1.  $\tilde{E} : \{p : \text{indexdeg}_{[1, n] \setminus I}(p) \leq n'\} \rightarrow \mathbb{R}$  is a linear map which is symmetric under permutations of  $[1, n] \setminus I$
2. For all  $i$  and all polynomials  $f$  such that  $\text{indexdeg}_{[1, n] \setminus I}(f) + \text{indexdeg}_{[1, n] \setminus I}(s_i) \leq n'$ ,  $\tilde{E}[fs_i] = 0$
3. For all polynomials  $f, g$  which are symmetric under permutations of  $[1, n] \setminus I$  such that  $\text{indexdeg}_{[1, n] \setminus I}(f) + \text{indexdeg}_{[1, n] \setminus I}(g) \leq n'$ ,  $\tilde{E}[fg] = \tilde{E}[f]\tilde{E}[g]$ .

We say that  $S$  is a level  $n'$  single-graph mimic for  $(P, A)$  if  $S$  is a self-consistent level  $n'$  story for  $(P, A)$  and  $\tilde{E}_S$  is a level  $n'$  single-graph mimic for  $P$  on  $[1, n] \setminus I_A$ .

A third property we want is that is that our story assigns non-negative probabilities to its substories as long as we don't query too many indices. If our story and all of its substories satisfy these three properties then we call it a good story.

► **Definition 33.** We say that  $S$  is a level  $(r, n')$  good story for  $(P, A)$  if the following conditions hold:

1.  $S$  is a level  $n'$  single graph mimic for  $(P, A)$ .
2. If  $r > 0$  then for any  $i \in [1, n] \setminus I_A$ , the values  $p_{ij}$  are non-negative and the stories  $\{S_{ij}\}$  are all level  $(r - 1, n' - 1)$  good stories for  $(P, A \cup (i))$ .

### 5.3 SOS lower bounds from good stories

We now prove that good stories imply SOS lower bounds.

► **Theorem 34.** *Let  $P$  be a symmetric problem with equations  $\{s_i = 0\}$ . If we have a level  $(r, n')$  good story for  $P$  then index degree  $d = \min\{2r, n'\}$  SOS fails to refute the equations for  $P$ .*

**Proof.** We need two components to prove this theorem. The first component is the following theorem which shows that if we have a good story then we satisfy all of the linear constraints on  $\tilde{E}$  and we have that  $\tilde{E}[g^2] \geq 0$  whenever  $g$  is symmetric under permutations of all but a few indices.

► **Theorem 35.** *Let  $P$  be a symmetric graph problem with constraints  $\{s_i = 0\}$  (where the  $\{s_i\}$  are polynomials in the input variables). If we have a level  $(r, n')$  good story  $S$  for  $P$  then the corresponding linear map  $\tilde{E}_S : \{f : \text{indexdeg}(f) \leq n'\} \rightarrow \mathbb{R}$  satisfies the following properties*

1.  $\tilde{E}_S$  is symmetric under permutations of  $[1, n]$
2. If  $I \subseteq [1, n]$  is a subset of indices of size at most  $r$  and  $g$  is a polynomial which is symmetric under permutations of  $[1, n] \setminus I$  such that  $\text{indexdeg}_{[1, n] \setminus I}(g) \leq \frac{n' - |I|}{2}$  then  $\tilde{E}_S[g^2] \geq 0$
3. For all  $i$  and all  $f$  such that  $\text{indexdeg}(f) + \text{indexdeg}(s_i) \leq n'$ ,  $\tilde{E}_S[fs_i] = 0$ .

## 61:14 Sum of Squares Lower Bounds from Symmetry and a Good Story

**Proof.** Since  $S$  is a single graph mimic and single graph mimics are symmetric with respect to permutations of  $[1, n]$ , the first statement follows. Similarly, the third statement follows directly from condition 2 of Definition 32

For the second statement, by conditions 1 and 2 of Definition 33, we can express  $\tilde{E}_S$  as a probability distribution  $\Omega$  over level  $n - |I|$  single graph mimics  $\tilde{E}_j$  for  $P$  on  $[1, n] \setminus I$ . Since  $g$  is symmetric under permutations of  $[1, n] \setminus I$ , for all of the  $\tilde{E}_j$ ,  $\tilde{E}_j[g^2] = \tilde{E}_j[g]\tilde{E}_j[g] \geq 0$ . We now have that  $\tilde{E}_S[g^2] = E_{E_j \sim \Omega} [\tilde{E}_j[g^2]] \geq 0$ , as needed.  $\blacktriangleleft$

The second component we need is Theorem 21, which shows that it is sufficient to verify that  $\tilde{E}_S[g^2] \geq 0$  whenever  $g$  is symmetric with respect to permutations of all but a few indices. which is exactly what is shown by Theorem 35.

With these components in hand, we now prove Theorem 34. We need to check the following:

1. Whenever  $\text{indexdeg}(f) + \text{indexdeg}(s_i) \leq d = \min\{2r, n'\}$ ,  $\tilde{E}_S[fs_i] = 0$ .
2. Whenever  $\text{indexdeg}(g) \leq \frac{d}{2} = \min\{r, \frac{n'}{2}\}$ ,  $\tilde{E}_S[g^2] \geq 0$

For the first statement, note that  $\text{indexdeg}(f) + \text{indexdeg}(s_i) \leq n'$ , so by Theorem 35,  $\tilde{E}_S[fs_i] = 0$ . For the second statement, given a polynomial  $g$  of index degree at most  $\frac{d}{2}$ , by Theorem 21 we can write

$$\tilde{E}_S[g^2] = \sum_{I \subseteq [1, n], j: |I| \leq \text{indexdeg}(g)} \tilde{E}_S[g_{I_j}^2]$$

where for all  $I, j$ ,

$$\forall i \in I, \quad \sum_{\sigma \in S_{[1, n] \setminus (I \setminus \{i\})}} \sigma(g_{I_j}) = 0$$

We now use the following lemma to upper bound  $\text{indexdeg}_{[1, n] \setminus I}(g_{I_j})$ :

► **Lemma 36.** *If  $g_{I_j}$  is symmetric with respect to permutations of  $[1, n] \setminus I$  and*

$$\forall i \in I, \quad \sum_{\sigma \in S_{[1, n] \setminus (I \setminus \{i\})}} \sigma(g_{I_j}) = 0$$

*then all monomials in  $g_{I_j}$  depend on all of the indices in  $I$*

**Proof.** Assume that there is an  $i \in I$  and some monomial  $p$  which does not depend on  $i$  which has a nonzero coefficient in  $g_{I_j}$ . By symmetry, for all permutations  $\sigma$  of  $[1, n] \setminus I$ , the coefficient of  $\sigma(p)$  is the same as the coefficient of  $p$ . However, these are also the coefficients of  $\sigma_2(p)$  for permutations  $\sigma_2$  of  $[1, n] \setminus (I \setminus \{i\})$ . Since  $\forall i \in I, \sum_{\sigma \in S_{[1, n] \setminus (I \setminus \{i\})}} \sigma(g_{I_j}) = 0$ , all of these coefficients must be 0, which is a contradiction.  $\blacktriangleleft$

This lemma implies that for all of the  $g_{I_j}$ ,  $\text{indexdeg}_{[1, n] \setminus I}(g_{I_j}) \leq \frac{n'}{2} - |I| \leq \frac{n' - |I|}{2}$ . Thus, by Theorem 35,  $\tilde{E}_S[g_{I_j}^2] \geq 0$ . Since this holds for all  $I, j$ ,  $\tilde{E}_S[g^2] \geq 0$ , as needed.  $\blacktriangleleft$

## 6 Verifying good stories

In this section, we describe a method to verify that a story  $S$  is a good story. For this method, we make the following assumption.

► **Definition 37.** We assume that the problem equations and  $S$  depend on a set of parameters and we take  $\alpha_1, \dots, \alpha_m$  to be these parameters.

► **Remark.** For knapsack and the triangle problem, we have two parameters  $n$  and  $k$ . For the MOD 2 principle we only have the parameter  $n$ .



## 6.1 Useful story properties part 2

We now describe two additional properties our stories may have which are useful for verifying that they are good stories. Once the definitions are understood, these properties are generally recognizable on sight.

One property  $S$  usually has is that the linear maps  $\tilde{E}_{S,B}$  assign values to monomials which are rational functions of the parameters  $\alpha_1, \dots, \alpha_m$ .

► **Definition 38.** We say that a level  $n'$  story  $S$  for  $(P, A)$  is rational if the following conditions hold

1. For all  $B$  such that  $A \subseteq B$  and  $|I_B \setminus I_A| \leq n'$ , for all monomials  $p$  such that  $I(p) \subseteq I_B$ ,  $\tilde{E}_{S,B}[p]$  is a rational function of the parameters  $\alpha_1, \dots, \alpha_m$ .
2. The rational functions  $\{\tilde{E}_{S,B}[p] : A \subseteq B, |I_B \setminus I_A| \leq n', I(p) \subseteq I_B\}$  have a common denominator  $q_S(\alpha_1, \dots, \alpha_m)$  and the degree of the numerator is bounded by a function of  $n'$  and  $\text{indexdeg}(p)$ .

A second property our stories may have is that there are many settings of the parameters  $\alpha_1, \dots, \alpha_m$  for which  $S$  and  $\tilde{E}_S$  actually correspond to probabilities and expected values of the uniform distribution over permutations of a single input  $G_0$ .

► **Definition 39.** Let  $S$  be a story for  $(P, A)$

1. We say that  $S$  is honest for  $(\alpha_1, \dots, \alpha_m)$  if  $S$  corresponds to what happens if we take the uniform distribution for all permutations of an actual input graph  $G_0$  over  $[1, n] \setminus I_A$  and  $G_0$  satisfies the equations for  $P$ . Note that if this is the case then  $S$  is automatically a single graph mimic for  $(P, A)$  for the parameter values  $(\alpha_1, \dots, \alpha_m)$  and  $\tilde{E}_S[p] = E_{\sigma \in S_{[1,n] \setminus I_A}}[p(\sigma(G_0))]$
2. We say that  $S$  is  $z$ -honest for  $(\alpha_1, \dots, \alpha_{m-1})$  if there are at least  $z$  values of  $\alpha_m$  such that  $S$  is honest for  $(\alpha_1, \dots, \alpha_m)$ .
3. For all  $j \in [1, m-2]$ , we say that  $S$  is  $z$ -honest for  $(\alpha_1, \dots, \alpha_j)$  if there are at least  $z$  values of  $\alpha_{j+1}$  such that  $S$  is  $z$ -honest for  $(\alpha_1, \dots, \alpha_{j+1})$ .
4. We say that  $S$  is  $z$ -honest if there are at least  $z$  values of  $\alpha_1$  such that  $S$  is  $z$ -honest for  $(\alpha_1)$ .

The intuition is that it is difficult for SOS to determine whether the parameters take one of these values for which we actually have a solution or we are in between these values.

The following lemma is very useful

► **Lemma 40.** *Let  $S$  be a story which is  $z$ -honest. If  $p(\alpha_1, \dots, \alpha_m)$  is a polynomial such that  $\text{deg}(p) < z$  and  $p(\alpha_1, \dots, \alpha_m) = 0$  whenever  $S$  is honest for  $(\alpha_1, \dots, \alpha_m)$  then  $p(\alpha_1, \dots, \alpha_m) = 0$*

**Proof.** We prove this lemma by induction. Assume that  $p(\alpha_1, \dots, \alpha_m) = 0$  whenever  $S$  is  $z$ -honest for  $\alpha_1, \dots, \alpha_j$ .

Consider  $p$  as a polynomial in the variables  $\alpha_{j+1}, \dots, \alpha_m$ . Each monomial has a coefficient which is a polynomial  $c(\alpha_1, \dots, \alpha_j)$  and we must have that  $c(\alpha_1, \dots, \alpha_j) = 0$  whenever  $S$  is  $z$ -honest for  $\alpha_1, \dots, \alpha_j$ . We now show that all of these coefficients  $c(\alpha_1, \dots, \alpha_j)$  must be 0 whenever  $S$  is  $z$ -honest for  $\alpha_1, \dots, \alpha_{j-1}$ . To see this, consider such a polynomial  $c(\alpha_1, \dots, \alpha_j)$  and assume that we have  $\alpha_1, \dots, \alpha_{j-1}$  such that  $S$  is  $z$ -honest for  $\alpha_1, \dots, \alpha_{j-1}$ . Considering  $c$  as a polynomial in  $\alpha_j$ ,  $c(\alpha_j) = 0$  whenever  $S$  is  $z$ -honest for  $\alpha_1, \dots, \alpha_j$ , which by definition happens for at least  $z$  values of  $\alpha_j$ . Since  $\text{deg}(c) < z$ , we must have that  $c(\alpha_1, \dots, \alpha_j) = c(\alpha_j) = 0$ . Thus,  $p(\alpha_1, \dots, \alpha_m) = 0$  whenever  $S$  is  $z$ -honest for  $\alpha_1, \dots, \alpha_{j-1}$ , as needed. ◀

## 6.2 Sufficient conditions for single graph mimics

With these definitions, we can now give sufficient conditions for showing that a story  $S$  is a single graph mimic.

► **Lemma 41.** *Let  $S$  be a level  $n'$  story for  $(P, A)$ . If  $S$  and the parameter values  $\alpha_1, \dots, \alpha_m$  satisfy the following conditions*

1.  $S$  is rational and symmetric with respect to permutations of  $[1, n] \setminus I_A$ .
2. For all  $z > 0$ ,  $S$  is  $z$ -honest.
3. Letting  $q_S(\alpha_1, \dots, \alpha_m)$  be the common denominator for  $\{\tilde{E}_{S,B}[p] : A \subseteq B, |I_B \setminus I_A| \leq n', I(p) \subseteq I_B\}$ ,  $q_S(\alpha_1, \dots, \alpha_m) \neq 0$

then for the parameter values  $\alpha_1, \dots, \alpha_m$ ,  $S$  is a level  $n'$  single graph mimic for  $(P, A)$ .

**Proof.** We need to verify the following for the given values of  $\alpha_1, \dots, \alpha_m$ :

1.  $S$  is self-consistent.
2. For all  $i$  and all polynomials  $f$  such that  $\text{indexdeg}_{[1,n] \setminus I_A}(f) + \text{indexdeg}_{[1,n] \setminus I_A}(s_i) \leq n'$ ,  $\tilde{E}_S[fs_i] = 0$
3. For any polynomials  $f, g$  such that  $f, g$  are symmetric under permutations of  $[1, n] \setminus I_A$  and  $\text{indexdeg}_{[1,n] \setminus I_A}(f) + \text{indexdeg}_{[1,n] \setminus I_A}(g) \leq n'$ ,  $\tilde{E}_S[fg] = \tilde{E}_S[f]\tilde{E}_S[g]$ .

We first verify that  $S$  is self-consistent for the given values of  $\alpha_1, \dots, \alpha_m$ . Let  $p$  be a monomial and let  $B, B'$  be sequences of indices such that  $A \subseteq B$ ,  $A \subseteq B'$ , and  $I(p) \subseteq I_B \cap I_{B'}$ . Since  $S$  is rational,  $\tilde{E}_{S,B}[p] = \frac{p_1(\alpha_1, \dots, \alpha_m)}{q(\alpha_1, \dots, \alpha_m)}$  and  $\tilde{E}_{S,B'}[p] = \frac{p_2(\alpha_1, \dots, \alpha_m)}{q(\alpha_1, \dots, \alpha_m)}$  are rational functions of the parameters  $\alpha_1, \dots, \alpha_m$ . Now note that whenever  $S$  is honest for  $(\alpha_1, \dots, \alpha_m)$ ,  $\tilde{E}_{S,B}[p] = \tilde{E}_{S,B'}[p]$  which implies that

$$p_1(\alpha_1, \dots, \alpha_m)q_S(\alpha_1, \dots, \alpha_m) = p_2(\alpha_1, \dots, \alpha_m)q_S(\alpha_1, \dots, \alpha_m)$$

Since  $S$  is  $z$ -honest for all  $z > 0$ , by Lemma 40 we have that  $p_1q_S = p_2q_S$  as polynomials in  $\alpha_1, \dots, \alpha_m$ . Plugging in our actual values of  $\alpha_1, \dots, \alpha_m$ ,  $q_S(\alpha_1, \dots, \alpha_m) \neq 0$  so  $p_1(\alpha_1, \dots, \alpha_m) = p_2(\alpha_1, \dots, \alpha_m)$  and thus  $\tilde{E}_{S,B'}[p] = \tilde{E}_{S,B}[p]$ , as needed.

We can use similar ideas to prove the second and third statements but there is a subtle point we must be careful of. A problem equations  $s_i$  may be a polynomial which is symmetric in  $n \setminus I_A$  rather than being a fixed polynomial. In this case,  $\tilde{E}_S[s_i]$  and  $\tilde{E}_S[fs_i]$  will still be rational functions in the parameters  $\alpha_1, \dots, \alpha_m$ . However, the equality  $\tilde{E}_S[fs_i] = \frac{p_{fs_i}(\alpha_1, \dots, \alpha_m)}{q_S(\alpha_1, \dots, \alpha_m)}$  may break down if

$$\text{indexdeg}_{[1,n] \setminus I_A}(f) + \text{indexdeg}_{[1,n] \setminus I_A}(s_i) > n'$$

► **Example 42.** If  $f = x_1x_2$  and  $s_i = \sum_{i=1}^n x_i - k$  then

$$fs_i = x_1^2x_2 + x_1x_2^2 + x_1x_2 \sum_{i \in [1,n] \setminus \{1,2\}} x_i - kx_1x_2$$

and by symmetry

$$\tilde{E}_S[fs_i] = \tilde{E}_S[x_1^2x_2] + \tilde{E}_S[x_1x_2^2] + (n-2)\tilde{E}_S[x_1x_2x_3] - k\tilde{E}_S[x_1x_2]$$

Thus,  $fs_i$  generally has index degree 3 and  $\tilde{E}_S[fs_i] = \frac{p_{fs_i}(\alpha_1, \dots, \alpha_m)}{q_S(\alpha_1, \dots, \alpha_m)}$  is a rational function of the parameters  $\alpha_1, \dots, \alpha_m$ . However, if  $n' = n = 2$  then we are missing the term  $x_1x_2 \sum_{i \in [1,n] \setminus \{1,2\}} x_i$  from  $fs_i$  which may break the equality  $\tilde{E}_S[fs_i] = \frac{p_{fs_i}(\alpha_1, \dots, \alpha_m)}{q_S(\alpha_1, \dots, \alpha_m)}$ . Note that this problem will not occur as long as

$$\text{indexdeg}_{[1,n] \setminus I_A}(f) + \text{indexdeg}_{[1,n] \setminus I_A}(s_i) \leq n'$$

With this point in mind, for the second statement, note that since  $S$  is rational and  $\text{indexdeg}(f) + \text{indexdeg}(s_i) \leq n'$ , we can write  $\tilde{E}_S[fs_i] = \frac{p_{fs_i}(\alpha_1, \dots, \alpha_m)}{q(\alpha_1, \dots, \alpha_m)}$ . Now observe that  $\tilde{E}[fs_i] = 0$  whenever  $\tilde{E}$  is honest for  $(\alpha_1, \dots, \alpha_m)$  and thus  $p_{fs_i}(\alpha_1, \dots, \alpha_m) = 0$  whenever  $S$  is honest for  $(\alpha_1, \dots, \alpha_m)$ . Since  $S$  is  $z$ -honest for all  $z > 0$ , by Lemma 40,  $p_{fs_i}(\alpha_1, \dots, \alpha_m) = 0$  as a polynomial. Plugging in the given values of  $\alpha_1, \dots, \alpha_m$ ,  $q(\alpha_1, \dots, \alpha_m) \neq 0$  so  $\tilde{E}_S[fs_i] = \frac{p_{fs_i}(\alpha_1, \dots, \alpha_m)}{q(\alpha_1, \dots, \alpha_m)} = 0$ , as needed.

Similarly, for the third statement we want to view  $f$ ,  $g$ , and  $fg$  as polynomials which depend on  $n$  rather than being fixed polynomials. Still, since  $S$  is rational and  $\text{indexdeg}(f) + \text{indexdeg}(g) \leq n'$ , we can write  $\tilde{E}_S[f] = \frac{p_f(\alpha_1, \dots, \alpha_m)}{q(\alpha_1, \dots, \alpha_m)}$ ,  $\tilde{E}_S[g] = \frac{p_g(\alpha_1, \dots, \alpha_m)}{q(\alpha_1, \dots, \alpha_m)}$ , and  $\tilde{E}_S[fg] = \frac{p_{fg}(\alpha_1, \dots, \alpha_m)}{q(\alpha_1, \dots, \alpha_m)}$ . Now observe that  $\tilde{E}_S[fg] = \tilde{E}_S[f]\tilde{E}_S[g]$  whenever  $S$  is honest for  $(\alpha_1, \dots, \alpha_m)$  and thus

$$p_f(\alpha_1, \dots, \alpha_m)p_g(\alpha_1, \dots, \alpha_m) - q(\alpha_1, \dots, \alpha_m)p_{fg}(\alpha_1, \dots, \alpha_m) = 0$$

whenever  $S$  is honest for  $(\alpha_1, \dots, \alpha_m)$ . Since  $S$  is  $z$ -honest for all  $z$ , by Lemma 40,  $p_f p_g - q p_{fg} = 0$  as a polynomial. Plugging in the given parameters  $\alpha_1, \dots, \alpha_m$ ,  $q(\alpha_1, \dots, \alpha_m) \neq 0$  so

$$\tilde{E}_S[fg] = \frac{p_{fg}(\alpha_1, \dots, \alpha_m)}{q(\alpha_1, \dots, \alpha_m)} = \frac{p_f(\alpha_1, \dots, \alpha_m)p_g(\alpha_1, \dots, \alpha_m)}{(q(\alpha_1, \dots, \alpha_m))^2} = \tilde{E}_S[f]\tilde{E}_S[g] \quad \blacktriangleleft$$

### 6.3 Verifying good stories

We are now ready to give sufficient conditions for a story to be a good story.

► **Theorem 43.** *If  $S$  is a story for  $(P, A)$  such that*

1.  $S$  is symmetric with respect to permutations of  $[1, n] \setminus I_A$
2.  $S$  is rational
3. For all  $z > 0$ ,  $S$  is  $z$ -honest.

*then for a given choice of parameters  $\alpha_1, \dots, \alpha_m$ , if  $n'$  and  $r$  are numbers such that  $n' \leq n - |I_A|$  and*

1. *If we consider up to  $r$  further indices, the probabilities  $p_{ij}$  are always non-negative.*
2. *If we consider up to  $n'$  further indices, we may get negative values for some  $p_{ij}$  but these values are always well-defined (i.e. the denominator is nonzero).*

*then  $S$  is a level  $(n', r)$  good story for  $(P, A)$ .*

**Proof.** Since we can consider up to  $n'$  further indices and get well-defined values for the  $p_{ij}$ ,  $S$  is a level  $n'$  story for  $(P, A)$ . Now by Lemma 41,  $S$  is a level  $n'$  single graph mimic for  $(P, A)$ .

We now prove the theorem by induction on  $r$ . The base case  $r = 0$  is trivial. If  $r > 0$  then for all  $i \in [1, n] \setminus I_A$ ,  $S$  gives non-negative values  $\{p_{ij}\}$  for the probabilities of level  $n' - 1$  stories  $S_{ij}$  for  $(P, A \cup (i))$ . Now note that for each of these  $S_{ij}$ , the values of subsequent  $p_{ij}$  will always be non-negative if we consider up to  $r - 1$  further indices and will be well-defined if we consider up to  $n' - 1$  further indices. Moreover,  $S_{ij}$  is symmetric with respect to permutations of  $[1, n] \setminus (I_A \cup \{i\})$ , rational, and is  $z$ -honest because  $S_{ij}$  is honest for  $(\alpha_1, \dots, \alpha_m)$  whenever  $S$  is honest for  $(\alpha_1, \dots, \alpha_m)$ . Thus, by the inductive hypothesis, each  $S_{ij}$  is a level  $(r - 1, n' - 1)$  good story for  $(P, A \cup (i))$  so  $S$  is a level  $(r, n')$  good story for  $(P, A)$ , as needed. ◀

## 6.4 Good stories for knapsack, the MOD 2 principle, and the triangle problem

In this subsection, we apply Theorem 43 to verify that our stories for knapsack, the MOD 2 principle, and the triangle problem are good stories.

► **Theorem 44.**

1. *Saying that we take  $k$  out of  $n$  elements is a level  $(\lfloor \min\{k, n - k\} \rfloor + 1, n)$  good story for the knapsack problem.*
2. *Saying that every vertex is incident with precisely one edge is a level  $(\lfloor \frac{n}{2} \rfloor + 1, n)$  good story for the MOD 2 principle.*
3. *Saying that we have  $k$  independent sets of size  $\frac{n}{k}$  is a level  $(\lfloor \min\{k, \frac{n}{k}\} \rfloor + 1, n)$  good story for the triangle problem.*

**Proof.** For knapsack and the triangle problem, we take  $\alpha_1 = n$  and  $\alpha_2 = k$ . For the MOD 2 principle, we just take  $\alpha_1 = n$ .

Our stories are clearly rational and symmetric with respect to permutations of  $[1, n]$ . We now check that they are  $z$ -honest for all  $z$ .

For knapsack, note that our story is honest for  $(n, k)$  whenever  $k$  is an integer between 0 and  $n$ . Thus, whenever  $n \geq z$  there are at least  $z$  values of  $k$  such that our story is honest for  $(n, k)$ , which implies that our story is  $z$ -honest for  $(n)$  whenever  $n \geq z$ . For all  $z$  there are infinitely many values of  $n$  such that  $n \geq z$  so our story is  $z$ -honest for all  $z$ , as needed.

For the MOD 2 principle, note that our story is honest for  $(n)$  whenever  $n$  is an even integer. There are infinitely many even integers so our story is  $z$ -honest for all  $z$ , as needed.

For the triangle problem, note that our story is honest for  $(n, k)$  whenever  $k$  is an integer and  $n$  is divisible by  $k$ . Thus, whenever  $n = a!$  and  $a \geq z$  then there are at least  $z$  values of  $k$  such that our description is honest for  $(n, k)$ , which implies that our story is  $z$ -honest for  $(n)$  whenever  $n = a!$  and  $a \geq z$ . For all  $z$  there are infinitely many values of  $n$  such that  $n = a!$  where  $a \geq z$  so our story is  $z$ -honest for all  $z$ , as needed.

All that we have to do now is to determine  $n'$  and  $r$ .

For knapsack, when we consider polynomials of index degree at most  $n'$ , the common denominator will be  $n(n - 1) \dots (n - n' + 1)$  as we are choosing  $n'$  elements one by one from  $[1, n]$ . This is well-defined as long as  $n' \leq n$  so we may take  $n' = n$ . The probabilities will be non-negative up to the  $(\lfloor \min\{k, n - k\} \rfloor + 1)$ -th index we consider, so we may take  $r = \lfloor \min\{k, n - k\} \rfloor + 1$ .

For the MOD 2 principle, when we consider polynomials of index degree at most  $n'$ , the common denominator will be  $n(n - 1) \dots (n - n' + 1)$  as we are choosing  $n'$  elements one by one from  $[1, n]$ . This is well-defined as long as  $n' \leq n$  so we may take  $n' = n$ . The probabilities will be non-negative up to the  $(\lfloor \frac{n}{2} \rfloor + 1)$ -th index we consider, so we may take  $r = \lfloor \frac{n}{2} \rfloor + 1$ .

For the triangle problem, when we consider polynomials of index degree at most  $n'$ , the common denominator will be  $k^{n'} n(n - 1) \dots (n - n' + 1)$ . The additional  $k^{n'}$  factor appears because there are  $\frac{n}{k}$  choices for the first element in an independent set of size  $\frac{n}{k}$ ,  $\frac{n-k}{k}$  choices for the second element, etc. Again, this is well-defined as long as  $n' \leq n$  so we may take  $n' = n$ . The probabilities will be non-negative up to the  $(\lfloor \min\{k, \frac{n}{k}\} \rfloor + 1)$ -th index we consider, so we may take  $r = \lfloor \min\{k, \frac{n}{k}\} \rfloor + 1$  ◀

► **Corollary 45.**

1. *For all positive integers  $n$  and all non-integer  $k \in [0, n]$ , index degree  $\min\{2\lfloor \min\{k, n - k\} \rfloor + 3, n\}$  SOS fails to refute the knapsack equations.*

2. For all odd  $n$ , index degree  $n$  SOS fails to refute the equations for the MOD 2 principle.
3. For all  $n \geq 6$ , and all  $k \in [1, n]$  such that  $k \notin \mathbb{Z}$  or  $\frac{n}{k} \notin \mathbb{Z}$ , index degree  $2\lfloor \min\{k, \frac{n}{k}\} \rfloor + 2$  SOS fails to refute the claim that Goodman's bound can be achieved for the triangle problem.

---

**References**


---

- 1 Sanjeev Arora, Boaz Barak, and David Steurer. Subexponential Algorithms for Unique Games and Related Problems. *J. ACM*, 62(5):42:1–42:25, 2015.
- 2 Sanjeev Arora, Satish Rao, and Umesh Vazirani. Expander Flows, Geometric Embeddings and Graph Partitioning. *J. ACM*, 56(2):5:1–5:37, April 2009.
- 3 Boaz Barak, Siu On Chan, and Pravesh Kothari. Sum of Squares Lower Bounds from Pairwise Independence. In *Proceedings of the Forty-seventh Annual ACM Symposium on Theory of Computing, STOC '15*, pages 97–106, 2015.
- 4 Boaz Barak, Samuel Hopkins, Jonathan Kelner, Pravesh Kothari, Ankur Moitra, and Aaron Potechin. A Nearly Tight Sum-of-Squares Lower Bound for the Planted Clique Problem. In *IEEE 57th Annual Symposium on Foundations of Computer Science, FOCS 2016, 9-11 October 2016, Hyatt Regency, New Brunswick, New Jersey, USA*, pages 428–437, 2016.
- 5 Boaz Barak, Jonathan A. Kelner, and David Steurer. Rounding Sum-of-squares Relaxations. In *Proceedings of the Forty-sixth Annual ACM Symposium on Theory of Computing, STOC '14*, pages 31–40, 2014.
- 6 Boaz Barak, Jonathan A. Kelner, and David Steurer. Dictionary Learning and Tensor Decomposition via the Sum-of-Squares Method. In *Proceedings of the Forty-seventh Annual ACM Symposium on Theory of Computing, STOC '15*, pages 143–151, 2015.
- 7 Boaz Barak, Pravesh K. Kothari, and David Steurer. Quantum Entanglement, Sum of Squares, and the Log Rank Conjecture. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2017*, pages 975–988, 2017.
- 8 Boaz Barak and Ankur Moitra. Noisy Tensor Completion via the Sum-of-Squares Hierarchy. In *Proceedings of the 29th Conference on Learning Theory, COLT 2016, New York, USA, June 23-26, 2016*, pages 417–445, 2016.
- 9 Grigoriy Blekherman, João Gouveia, and James Pfieffer. Sum of Squares on the hypercube. *Mathematische Zeitschrift*, 284(1-2):41–54, 2016.
- 10 Yash Deshpande and Andrea Montanari. Improved Sum-of-Squares Lower Bounds for Hidden Clique and Hidden Submatrix Problems. In *COLT*, volume 40 of *JMLR Workshop and Conference Proceedings*, pages 523–562. JMLR.org, 2015.
- 11 Karin Gatermann and Pablo Parrilo. Symmetry groups, semidefinite programs, and sums of squares. *Journal of Pure and Applied Algebra*, 192(1-3):95–128, 2004.
- 12 Rong Ge and Tengyu Ma. Decomposing Overcomplete 3rd Order Tensors using Sum-of-Squares Algorithms. In *APPROX-RANDOM*, volume 40 of *LIPICs*, pages 829–849. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2015.
- 13 Michel X. Goemans and David P. Williamson. Improved Approximation Algorithms for Maximum Cut and Satisfiability Problems Using Semidefinite Programming. *J. ACM*, 42(6):1115–1145, November 1995.
- 14 A. Goodman. On sets of acquaintances and strangers at any party. *The American Mathematical Monthly*, 66(9):778–783, 1959.
- 15 Dima Grigoriev. Complexity of Positivstellensatz proofs for the knapsack. *Computational Complexity*, 10(2):139–154, 2001.
- 16 Dima Grigoriev. Linear lower bound on degrees of Positivstellensatz calculus proofs for the parity. *Theor. Comput. Sci.*, 259(1-2):613–622, 2001.

- 17 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 Trans. Algorithms*, 14(3):28:1–28:31, June 2018.
- 18 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. *CoRR*, abs/1710.05017, 2017.
- 19 Samuel B. Hopkins, Tselil Schramm, Jonathan Shi, and David Steurer. Fast spectral algorithms from sum-of-squares proofs: tensor decomposition and planted sparse vectors. In *STOC*, pages 178–191, 2016.
- 20 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*, STOC 2017, pages 132–145, 2017.
- 21 Adam Kurpisz, Samuli Leppänen, and Monaldo Mastrolilli. Sum-of-Squares Hierarchy Lower Bounds for Symmetric Formulations. In *IPCO*, volume 9682 of *Lecture Notes in Computer Science*, pages 362–374. Springer, 2016.
- 22 Jean B. Lasserre. Global Optimization with Polynomials and the Problem of Moments. *SIAM J. on Optimization*, 11(3):796–817, March 2000.
- 23 Monique Laurent. Lower Bound for the Number of Iterations in Semidefinite Hierarchies for the Cut Polytope. *Math. Oper. Res.*, 28(4):871–883, 2003.
- 24 Troy Lee, Anupam Prakash, Ronald de Wolf, and Henry Yuen. On the Sum-of-squares Degree of Symmetric Quadratic Functions. In *Proceedings of the 31st Conference on Computational Complexity*, CCC ’16, pages 17:1–17:31, 2016.
- 25 Tengyu Ma, Jonathan Shi, and David Steurer. Polynomial-Time Tensor Decompositions with Sum-of-Squares. In *FOCS*, pages 438–446. IEEE Computer Society, 2016.
- 26 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*, STOC ’15, pages 87–96, 2015.
- 27 Yuri Nesterov. Squared functional systems and optimization problems. *High Performance Optimization*, pages 405–440, 2000.
- 28 Pablo Parrilo. *Structured semidefinite programs and semialgebraic geometry methods in robustness and optimization*. PhD thesis, California Institute of Technology, 2000.
- 29 Aaron Potechin and David Steurer. Exact tensor completion with sum-of-squares. In *Proceedings of the 30th Conference on Learning Theory, COLT 2017, Amsterdam, The Netherlands, 7-10 July 2017*, pages 1619–1673, 2017.
- 30 Annie Raymond, James Saunderson, Mohit Singh, and Rekha R. Thomas. Symmetric sums of squares over  $k$ -subset hypercubes. *Math. Program.*, 167(2):315–354, 2018.
- 31 Alexander A. Razborov. Flag algebras. *J. Symb. Log.*, 72(4):1239–1282, 2007.
- 32 Grant Schoenebeck. Linear Level Lasserre Lower Bounds for Certain  $k$ -CSPs. In *FOCS*, pages 593–602. IEEE Computer Society, 2008.
- 33 N. Shor. An approach to obtaining global extremums in polynomial mathematical programming problems. *Cybernetics and Systems Analysis*, 23(5):695–700, 1987.
- 34 Madhur Tulsiani. CSP gaps and reductions in the lasserre hierarchy. In *Proceedings of the 41st Annual ACM Symposium on Theory of Computing, STOC 2009, Bethesda, MD, USA, May 31 - June 2, 2009*, pages 303–312, 2009.



# Learning Time Dependent Choice

**Zachary Chase**

Department of Mathematics, California Institute of Technology, Pasadena, USA  
zchase@caltech.edu

**Siddharth Prasad**

Department of Computing and Mathematical Sciences, California Institute of Technology,  
Pasadena, USA  
sprasad@caltech.edu

---

## Abstract

We explore questions dealing with the learnability of models of choice over time. We present a large class of preference models defined by a structural criterion for which we are able to obtain an exponential improvement over previously known learning bounds for more general preference models. This in particular implies that the three most important discounted utility models of intertemporal choice – exponential, hyperbolic, and quasi-hyperbolic discounting – are learnable in the PAC setting with VC dimension that grows logarithmically in the number of time periods. We also examine these models in the framework of active learning. We find that the commonly studied stream-based setting is in general difficult to analyze for preference models, but we provide a redeeming situation in which the learner can indeed improve upon the guarantees provided by PAC learning. In contrast to the stream-based setting, we show that if the learner is given full power over the data he learns from – in the form of learning via membership queries – even very naive algorithms significantly outperform the guarantees provided by higher level active learning algorithms.

**2012 ACM Subject Classification** Theory of computation → Models of learning

**Keywords and phrases** Intertemporal Choice, Discounted Utility, Preference Recovery, PAC Learning, Active Learning

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2019.62

**Acknowledgements** We would like to thank Federico Echenique and Adam Wierman for several helpful comments and suggestions.

## 1 Introduction

We study the learnability of economic models of choice over time. Our setting is that of an analyst who first observes an agent’s choices between plans that specify payoffs over time, and then attempts to learn the preference parameters guiding the choices. While such parameters are stylized – in reality subjects are not likely to perform standardized computations according to private parameters before making decisions – experiments have shown that they often provide accurate descriptions of how an agent behaves. By observing enough choice data, one can hope to learn the economic parameters that most closely describe the agent’s preferences. Thus, learning theory provides an especially meaningful lens with which to view the theory of choice – it allows us to answer questions regarding the volume of data required to faithfully predict future decisions made by an observed agent. The overarching goal of this paper is to identify structural criteria that yield strong learnability results for preferences over time under different restrictions placed on the learner/analyst. The criteria we present captures a



© Zachary Chase and Siddharth Prasad;  
licensed under Creative Commons License CC-BY  
10th Innovations in Theoretical Computer Science (ITCS 2019).  
Editor: Avrim Blum; Article No. 62; pp. 62:1–62:19



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



large class of preference models that give the agent significant freedom in weighting decisions against time delays. In particular, it encompasses the most popular models of time dependent choice used by economists.

The main economic application of our results is in understanding the learnability of models of intertemporal choice. Intertemporal choice is what governs an agent’s decisions over several time periods. The most important models of intertemporal choice are discounted utility models, in which agents evaluate plans by discounting actions as they are delayed – in analogy to how markets value the loss or gain of money over time. The first axiomatic treatment of discounting was by Koopmans in 1960 [18], in which he demonstrates that simple postulates for preferences over an infinite time horizon yield “impatience”. The three most commonly studied discounting models are *exponential*, *hyperbolic*, and *quasi-hyperbolic*, and all have been studied by both economists and computer scientists (though less so by the latter) as well as researchers from various other fields. The importance of discounted utility in economics cannot be overstated – it is the canonical framework used by economists to study choice over time.

Problems of learning economic parameters have received recent attention from computer scientists; see, e.g., [1, 2, 3, 16, 21]. Inspired by a general theme of demanding computational robustness from economic models (Echenique, Golovin, and Wierman provide a nice discussion of this topic in [11]), the tools of learning theory provide relevant and exciting perspectives from which to view economic models that have been around for several decades. In contrast to the usual goal of truthfully extracting the agent’s parameters adopted by classical mechanism design, the learning problem aims to efficiently extract a truthful agent’s parameters in the restricted message space of binary classification. Our paper contributes to the line of work that specifically studies models of choice using the perspectives of learning theory. This confluence of decision theory and learning theory was initiated by Basu and Echenique [2], who consider the learning problem for models of choice under uncertainty. Our investigation in this paper is motivated by models of how agents make choices over time. We provide learnability results that are fine tuned to structural requirements on such models.

We now summarize our main contributions at a high level. Section 3 contains a more detailed exposition of our results.

## Summary of results and techniques

Our situation is one of an analyst trying to learn the parameters governing an agent’s preferences over time. The two main learning themes we consider are (1) when the analyst has no control over the data he sees and (2) when the analyst has some control over the data he sees. The first theme is aptly captured by *probably approximately correct (PAC) learning*. To analyze the second theme, we investigate two models of *active learning*: stream-based selective sampling and membership queries.

In the first part, we study the PAC model, where the analyst is presented with pairs of alternatives and a label for each pair indicating the agent’s preference between the alternatives. The data points are drawn according to some unknown distribution, and the analyst has no control over the data he is presented with. Our main result here is a structural criterion on preference models that allows for a drastic improvement over the PAC learning complexity bounds achieved in [2]. We stipulate that the agent weights time-delayed payoffs according to polynomials, which allows for considerable freedom in how payoffs are weighted. Under this requirement, we show that such classes of preference models admit an exponential improvement in sample complexity bounds over the more general preference models considered in [2]. This is achieved via a computation of the VC dimension (which

quantifies the complexity of PAC learning). A simple application of our result shows that each of the discounted utility models are learnable (and in fact admit polynomial time learning algorithms), with sample complexity that grows logarithmically in the number of time periods  $T$  over which decisions are being made. The computation of the VC dimension is due to a natural connection between pairs of choices and the signs of polynomials that arise from the choices.

In the second part, we consider active learning models, where the analyst is given a certain amount of control over the data that he uses to learn. The two active learning models we study are *stream-based selective sampling* and learning via *membership queries*. In the former, the analyst is given some control over what data he learns from: as in the PAC setting he is presented with points drawn from an unknown distribution, but now the analyst chooses whether or not to see the label representing the agent's choice for each point. In the latter, the analyst has complete control over the data he learns from: the analyst can at any time request the label for any point. The former model seems to have been commonly adopted in order to study the very general problem of concept learning, when there is no extra information about the structure of the concepts. We find that the *disagreement methods* used to study the stream-based setting are in general difficult to analyze in the context of preference models – requiring quantitative information about the underlying distribution from which points are drawn. However, we provide a redeeming situation (by examining a particular distribution) where we obtain an improvement over the PAC guarantees. Membership queries, on the other hand, allow us to heavily exploit the structure of the preference models we consider. We present a naive membership query-based algorithm that significantly outperforms the guarantees provided in the stream-based setting. Learning via membership queries, we conclude, seems to be the appropriate model to actively learn economic parameters. It allows the analyst to make use of the preference relations' structure, and also precisely captures the situation in which the analyst and agent are participating in a real time experiment.

## Related work

Discounted utility models of intertemporal choice have been studied extensively not only by economists, but also by researchers from various other fields. We first briefly survey some of the relevant work pertaining to the exponential, quasi-hyperbolic, and hyperbolic discounting models and then survey existing work in the more general topic of learning economic parameters.

In the exponential discounting model, the agent evaluates his utilities based on a discount factor  $\delta \in (0, 1)$ , where a delay of  $t$  time periods incurs an exponential discount in utility by  $\delta^t$ . Climate change policies are traditionally evaluated according to an exponential discounting model – for example, the *Stern review* on the economics of climate change deals with issues of how to choose an appropriate discount rate in evaluating such policies [20]. Chambers and Echenique [8] present results related to the problem of aggregating discount rates proposed by a group of experts facing disagreement. While it is the most commonly used discounting model due to its simplicity, the exponential discounting model has been criticized due to its inability to match empirical data recording actual human behavior. Quasi-hyperbolic and hyperbolic discounting aim to mend such issues. The quasi-hyperbolic discounting model is parametrized by  $\beta, \delta \in (0, 1)$ , where a delay of  $t$  time periods incurs a discount in utility by  $\beta\delta^t$ , and was first introduced by Phelps and Pollack [19] to study preferences over generations. They proposed that the constant  $\beta$  discount factor represents how much a given generation  $t$  is affected by the utilities of other people relative to their own – and remark that  $\beta = 1$  represents “perfect altruism”, while  $\beta < 1$  represents “imperfect altruism”. Kleinberg and

Oren [17] study agents with quasi-hyperbolic discounting and propose a graph-theoretic model to investigate phenomena such as procrastination and abandonment of long-range tasks. Hyperbolic discounting aims to capture the notion that people are more impatient in making short term decisions (today vs. tomorrow) than long term decisions (365 days from today vs. 366 days from today)<sup>1</sup>, and is modeled via a discount of  $(1 + t\alpha)^{-1}$  at time  $t$ . Researchers in fields such as psychology and neuroscience [4, 15] have adopted the hyperbolic discounting model to study, for example, issues of self control and anticipation in humans and animals, and have compared the predictions by the different discounted utility models to neurobiological data obtained via MRI scans. Chabris et al. [7] give an exposition of the discounted utility models of intertemporal choice and survey sociological research that examines empirical data pertaining to how discount rates are affected by factors like age, drug use, gambling, etc.

The study of economic models has witnessed a recent influx of work from computer scientists dealing with questions of robustness under various notions of complexity (learning complexity, computational complexity, communication complexity, etc.). Kalai [16] in 2001 studied the learnability of choice functions, where the observed choices are in the form of a given set of alternatives along with the most preferred alternative from the set. Beigman and Vohra [3], Zadimoghaddam and Roth [21], and Balcan et al. [1] investigate the problem of learning utility functions in the context of an expected utility maximizing agent in a demand environment. Most recently (and most related to our work), Basu and Echenique [2] study the learnability of preference models of choice under uncertainty, in which an agent is uncertain about states of a lottery and is made to choose between acts that encode utilities over each state. Here, the different models of choice under uncertainty arise from different ways of representing the subjective probability held by an agent. They are also the first to study learnability in the decision-theoretic setting where choice is modeled by preference relations rather than by expected utility maximizing behavior in a demand setting. However, it does not appear that the learnability of models of intertemporal choice has been previously studied.

## 2 Model and Preliminaries

We now formally set up the discounted utility models of intertemporal choice and state the standard definitions from learning theory in the context of preference relations. Much of the following material regarding learning and preference relations is taken from [2] since we require a similar list of definitions and setup. First, we sketch our high level model.

Let  $X$  be a Euclidean space equipped with a Borel  $\sigma$ -algebra. A *preference relation* on  $X$  is a binary relation  $\succsim \subseteq X \times X$  such that  $\succsim$  is measurable with respect to the product  $\sigma$ -algebra on  $X \times X$ . A *model*  $\mathcal{P}$  of preference relations is a collection of preference relations.

An agent makes choices from pairs of alternatives  $(x^i, y^i)_{i=1}^n$  that are drawn according to some unknown distribution on  $X \times X$ . The choices are presented as labels  $(a_i)_{i=1}^n$  where  $a_i = 1$  if the agent chooses  $x^i$  and  $a_i = 0$  if the agent chooses  $y^i$ . A *dataset* is any finite sequence of pairs of plans and their labels  $((x^1, y^1), a_1), \dots, ((x^n, y^n), a_n)$ . An analyst observes a dataset, and attempts to guess the preference relation governing the agent's choices. A *learning rule* is any map  $\sigma$  from datasets to preference relations. The output of the learning rule is the analyst's hypothesis as to what the agent's true preference relation is, having seen some finite dataset.

---

<sup>1</sup> In particular note that exponential discounting does not capture this issue, i.e. it is *dynamically consistent*, in that preferences do not change according to shifts in time.

## 2.1 Learnability

The two notions of learnability we consider are the PAC model and the active model. We now state the standard definitions of PAC and active learning in the context of preference relations. Most of the following definitions for the PAC setting are taken from [2] since the setup involving preference relations is identical. These definitions of course apply to the more general setting of concept learning (for example, see [6]).

► **Definition 1.** A collection  $\mathcal{P}$  of preference relations is (PAC) learnable if there is a learning rule  $\sigma$  such that for every  $0 < \varepsilon, \delta < 1$ , there is  $s(\varepsilon, \delta) \in \mathbb{N}$  such that for every  $n \geq s(\varepsilon, \delta)$ ,  $\succsim \in \mathcal{P}$ , and  $\mu \in \Delta(X \times X)$ ,

$$\mu^n(\{(x^1, y^1), \dots, (x^n, y^n) : \mu(\succsim^* \Delta \succsim) > \varepsilon\}) < \delta,$$

where

$$\succsim^* = \sigma(\{(x^1, y^1), I_{x^1 \succsim y^1}, \dots, (x^n, y^n), I_{x^n \succsim y^n}\})$$

is the hypothesis preference relation produced by the learning rule<sup>2</sup>. The quantity  $s(\varepsilon, \delta)$  is called the *sample complexity* of the learning rule  $\sigma$ .

The complexity of learning is commonly quantified by the Vapnik-Chervonenkis (VC) dimension, defined based on so-called shattered sets. A set of points  $\{(x^1, y^1), \dots, (x^n, y^n)\}$  from  $X \times X$  is *shattered* by a model of preferences  $\mathcal{P}$  if for every vector of labels  $(a_1, \dots, a_n) \in \{0, 1\}^n$ , there is a preference relation  $\succsim \in \mathcal{P}$  that realizes the labelling, i.e. for  $i = 1, \dots, n$  we have that  $x^i \succsim y^i$  if and only if  $a_i = 1$ . In this case,  $\mathcal{P}$  is said to *rationalize* the dataset  $\{(x^1, y^1), a_1), \dots, ((x^n, y^n), a_n)\}$ . The VC dimension of  $\mathcal{P}$ , denoted by  $VC(\mathcal{P})$ , is the largest integer  $n$  such that there exist  $n$  points that are shattered by  $\mathcal{P}$ .

Blumer et al. [6] in 1989 proved that learnability is equivalent to having a finite VC dimension<sup>3</sup>.

► **Theorem 2.** A model of preferences  $\mathcal{P}$  is learnable if and only if  $VC(\mathcal{P}) < \infty$ .

The VC dimension (denoted by  $d$  for the remainder of this subsection) additionally characterizes the sample complexity of learning a model of preferences: the optimal sample complexity of PAC learning is  $s(\varepsilon, \delta) = \Theta\left(\frac{1}{\varepsilon} \left(d + \log \frac{1}{\delta}\right)\right)$  [14].

The other learning model we consider is the active learning framework, where the analyst has some control over the data from which he learns. In stream-based selective sampling, points drawn according to an unknown distribution are presented to the analyst as before, but without the labels. The analyst can choose whether or not to query the label of a given point, and the complexity of the learning rule is measured by *label complexity*, i.e. the number of labels requested by the analyst. Disagreement based active learning refers to the paradigm in which the learner only requests labels on points that significantly reduce the hypothesis space. The disagreement of a preference model with respect to the underlying distribution is quantified through the *disagreement coefficient*  $\theta$ , which is defined in Section 5. A finite disagreement coefficient implies (for the underlying distribution) an exponential improvement in label complexity over the sample complexity of PAC learning. For example, the CAL algorithm [9, 10, 13], a simple disagreement based learning algorithm, yields a label complexity of

$$\ell_{CAL}(\varepsilon, \delta) = O\left(\theta \log \frac{1}{\varepsilon} \left(d \log \theta + \log \frac{\log(1/\varepsilon)}{\delta}\right)\right).$$

<sup>2</sup>  $\mu^n$  denotes the product measure induced by  $\mu$  on  $(X \times X)^n$ .

<sup>3</sup> This result requires  $\mathcal{P}$  to satisfy a certain measurability requirement. We note in Section 4 that the models of choice we consider all satisfy said requirement.

In the membership queries model, the analyst has complete control over the learning data and is allowed to request the label for any point at any time. Learning via membership queries is thus a question of optimal algorithm design, and bounds on query complexity obtained in this manner would presumably be independent from learning-theoretic quantities (e.g. VC dimension and disagreement coefficient). There appears to be a dearth of literature/results pertaining to the complexity of membership query algorithms for learning when the hypothesis space is infinite. One explanation for this is that improvements to the “passive” disagreement based methods used in the stream-based setting would need specific information about the problem domain: disagreement based methods are designed to work on a very general class of concept learning problems without assuming anything about the learning space. In our case, we have specific details about how the preference relations take shape. Thus, the membership query model turns out to be an interesting and useful perspective to use in the study of learning preference models.

For a more detailed survey of active learning, see [10].

## 2.2 Discounted utility

We now present the definitions for the discounted utility models of intertemporal choice. An agent chooses between *plans* or vectors in  $X = \mathbb{R}^T$  that encode payoffs over  $T$  time periods. A preference relation over plans is a binary relation  $\succsim \subseteq \mathbb{R}^T \times \mathbb{R}^T$ .

The most important model of intertemporal choice is the *discounted utility model*, in which the agent’s payoffs  $x_t$  for having chosen a plan  $x \in \mathbb{R}^T$  are reduced, or discounted, as  $t$  increases from 1 to  $T$ . In its most general form, we can characterize the preference relations that follow time discounting as follows:

► **Definition 3** (Discounted utility model). The class of preference relations  $\mathcal{P}_{\mathcal{D}}$  that satisfy the *discounted utility model* are those  $\succsim$  such that there exists a decreasing map  $D : \{1, \dots, T\} \rightarrow (0, 1)$  where

$$x \succsim y \text{ if and only if } \sum_{t=1}^T D(t)x_t \geq \sum_{t=1}^T D(t)y_t.$$

We use the following notation for the preference models arising from the three most commonly studied discounting functions  $D$ :

- $\mathcal{P}_{\mathcal{D}}$  denotes the set of preferences that satisfy the discounted utility model.
- $\mathcal{P}_{\mathcal{ED}}$  denotes the set of preferences that satisfy the discounted utility model with *exponential discounting*:  $D(t) = \delta^t$  for  $\delta \in (0, 1)$ .
- $\mathcal{P}_{\mathcal{HD}}$  denotes the set of preferences that satisfy the discounted utility model with *hyperbolic discounting*:  $D(t) = \frac{1}{1+t\alpha}$  for  $\alpha > 0$ .
- $\mathcal{P}_{\mathcal{QHD}}$  denotes the set of preferences that satisfy the discounted utility model with *quasi-hyperbolic discounting*:  $D(t) = 1$  if  $t = 1$ ,  $D(t) = \beta \cdot \delta^{t-1}$  if  $t > 1$  for  $\beta, \delta \in (0, 1)$ .

For a more thorough exposition on the various discounted utility models of intertemporal choice, see [7].

## 3 Main Results

In this section we provide a formal discussion and interpretation of our results, which is split into two themes: the first dealing with an analyst who has no control over the learning data, the second dealing with an analyst who has some control over the learning data.

## A powerless analyst

The first part of our paper investigates the situation of an analyst trying to learn the preference relation by which an agent makes choices, but has no control over what choices he gets to observe, and is agnostic to the process by which they are drawn. We thus adopt the PAC learning model.

The agent chooses between plans that encode payoffs over  $T$  periods of time and evaluates the total payoff of a plan vector  $x \in \mathbb{R}^T$  according to private weights  $w_1, \dots, w_T$  that he multiplicatively applies to each state:  $\text{payoff}(x) = \sum_{t=1}^T w_t x_t$ . This defines a model of preference relations, which we denote by  $\mathcal{P}_{\mathcal{W}}$ , where for any  $\succsim \in \mathcal{P}_{\mathcal{W}}$ , there exists a vector of weights  $w = (w_1, \dots, w_T) \in \mathbb{R}^T$  such that

$$x \succsim y \text{ if and only if } w \cdot x \geq w \cdot y.$$

In [2], it is shown that  $T - 1 \leq VC(\mathcal{P}_{\mathcal{W}}) \leq T + 1$ . In the context of choice over time, however, this model is extremely general and does not capture any of the intuitive notions of how an agent values payoffs when they are delayed<sup>4</sup>. For example, the discounted utility models of intertemporal choice require the weights to be of a particular functional form. Moreover, when there is no structure to the discount function we cannot improve the bounds on  $\mathcal{P}_{\mathcal{W}}$ :

► **Proposition 4.**  $T - 1 \leq VC(\mathcal{P}_{\mathcal{D}}) \leq T + 1$ .

This leads us to the motivating question of the first part of the paper: what structural conditions can we impose on the weights  $w_1, \dots, w_T$  such that this bound can be improved?

We investigate the situation where the agent computes his weights by evaluating polynomials at a private parameter  $\delta$ . Specifically, let  $Q_1, \dots, Q_T$  be polynomials of degree at most  $d$ , and suppose the agent evaluates total payoff of a plan vector  $x \in \mathbb{R}^T$  by  $\text{payoff}(x) = \sum_{t=1}^T Q_t(\delta)x_t$ . Consequently, let  $\mathcal{P}_{\mathcal{P}\mathcal{W}}$  be the model of preference relations parametrized by  $\delta$  such that

$$x \succsim y \text{ if and only if } \sum_{t=1}^T Q_t(\delta)x_t \geq \sum_{t=1}^T Q_t(\delta)y_t.$$

This class of preference models allows us to approximate preference relations where the weights are given by any real valued functions – we choose  $Q_1, \dots, Q_T$  to be the appropriate Taylor polynomials. Moreover, existing models of intertemporal choice fit this characterization – for example  $\mathcal{P}_{\mathcal{E}\mathcal{D}}$  and  $\mathcal{P}_{\mathcal{H}\mathcal{D}}$ .

We additionally consider a slightly larger class of preference models where the agent has a private parameter  $\beta$  (in addition to  $\delta$ ) that in evaluating total payoff of a plan vector  $x \in \mathbb{R}^T$  allows the agent to modify the constant term  $\sum_{t=1}^T Q_t(0)x_t$  of the polynomial  $\sum_{t=1}^T Q_t(\delta)x_t$ . This model aims to more generally capture the effects of the  $\beta$  parameter in quasi-hyperbolic discounting. For polynomials  $Q_1, \dots, Q_T$  of degree at most  $d$ , let  $\mathcal{P}_{\mathcal{B}\mathcal{P}\mathcal{W}}$  be the model of preference relations parametrized by  $\beta$  and  $\delta$  such that  $x \succsim y$  if and only if

$$\left(\frac{1}{\beta} - 1\right) \sum_{t=1}^T Q_t(0)x_t + \sum_{t=1}^T Q_t(\delta)x_t \geq \left(\frac{1}{\beta} - 1\right) \sum_{t=1}^T Q_t(0)y_t + \sum_{t=1}^T Q_t(\delta)y_t.^5$$

<sup>4</sup> In [2] the complete control over weights is used to model choice under uncertainty, which calls for such generality since the agent's beliefs/weights are given by an element of the probability simplex on  $\mathbb{R}^T$ .

Our main results show that with this additional structure on the preference model, we can achieve an exponential improvement in the bounds for the VC dimension of  $\mathcal{P}_{\mathcal{W}}$  obtained in [2]<sup>6</sup>.

► **Theorem 5.** *For every  $\varepsilon > 0$ , there exists a  $d_\varepsilon$  such that for every  $d \geq d_\varepsilon$  we have  $VC(\mathcal{P}_{\mathcal{P}\mathcal{W}}), VC(\mathcal{P}_{\mathcal{B}\mathcal{P}\mathcal{W}}) \leq (1 + \varepsilon) \log d$  for any  $T$  and any  $T$  polynomials  $Q_1, \dots, Q_T$  of degree at most  $d$ .*

Note that when  $Q_1, \dots, Q_T$  have degree at most polynomial in  $T$ , we obtain an exponential improvement over the linear growth of  $VC(\mathcal{P}_{\mathcal{W}})$ . We show that in this case, we get a tight (asymptotic) bound of  $\log T$ :

► **Theorem 6.** *Let  $Q_1, \dots, Q_T$  be polynomials in  $\delta$  of degree at most  $T - 1$  that span the space of polynomials in  $\delta$  of degree at most  $T - 1$ . Then  $VC(\mathcal{P}_{\mathcal{P}\mathcal{W}}), VC(\mathcal{P}_{\mathcal{B}\mathcal{P}\mathcal{W}}) \geq \log(T - 1)$ .*

An interesting feature of Theorems 5 and 6 is that for fixed  $Q_1, \dots, Q_T$  with degrees at most  $T - 1$ ,  $VC(\mathcal{P}_{\mathcal{P}\mathcal{W}})$  and  $VC(\mathcal{P}_{\mathcal{B}\mathcal{P}\mathcal{W}})$  satisfy the same asymptotic bounds, so giving the agent an extra parameter that allows control over the constant term of the polynomial  $\sum_{t=1}^T Q_t(\delta)x_t$  does not introduce a significant amount of richness to the model.

Applying Theorems 5 and 6 to the discounted utility models, we have:

► **Corollary 7.**  $VC(\mathcal{P}_{\mathcal{E}\mathcal{D}}), VC(\mathcal{P}_{\mathcal{H}\mathcal{D}}), VC(\mathcal{P}_{\mathcal{Q}\mathcal{H}\mathcal{D}}) \sim \log(T - 1)$

Thus,  $\mathcal{P}_{\mathcal{D}}$ ,  $\mathcal{P}_{\mathcal{E}\mathcal{D}}$ ,  $\mathcal{P}_{\mathcal{H}\mathcal{D}}$ , and  $\mathcal{P}_{\mathcal{Q}\mathcal{H}\mathcal{D}}$  are all learnable.  $\mathcal{P}_{\mathcal{D}}$  requires a minimum sample size that grows linearly with  $T$ , while  $\mathcal{P}_{\mathcal{E}\mathcal{D}}$ ,  $\mathcal{P}_{\mathcal{H}\mathcal{D}}$ , and  $\mathcal{P}_{\mathcal{Q}\mathcal{H}\mathcal{D}}$  require a minimum sample size that grows logarithmically in  $T$ . We also note that these classes are all efficiently learnable, that is, the learning rule can be described by a polynomial time algorithm.

The main technique in proving Theorems 5 and 6 is interpreting the shattering criteria as a statement about the sign combinations achieved by a collection of polynomials. The upper bound on the VC dimension follows from an upper bound on the number of sign combinations a collection of polynomials can achieve. In demonstrating the lower bound on the VC dimension, we construct a set of points that is shattered by finding polynomials achieving all possible sign combinations – chosen according to a Hamiltonian path in the  $\log(T - 1)$ -dimensional hypercube.

Table 1 summarizes the definitions and bounds on the VC dimensions for the various preference models we consider.

## A powerful analyst

Our other results concern the active learning framework, which broadly deals with situations in which the analyst has some control over the choices he observes and learns from. The two models we consider are *stream-based selective sampling* and learning via *membership queries*. A large body of active learning research is devoted to the stream-based model, specifically focusing on disagreement based algorithms – a class of learning algorithms that instructs the analyst only to request labels on points he sees that reduce the hypothesis space significantly. In the most general setting of concept learning, this is a useful framework since the error guarantees can be described using the same setup as the PAC model. Moreover without additional information about the problem domain, it is unclear how to devise efficient algorithms that are more specific in instructing the analyst on what questions to ask.

<sup>6</sup> It is important to note that the classes  $\mathcal{P}_{\mathcal{P}\mathcal{W}}$  and  $\mathcal{P}_{\mathcal{B}\mathcal{P}\mathcal{W}}$  are defined for a given  $Q_1, \dots, Q_T$ . That is, the analyst knows  $Q_1, \dots, Q_T$ , and is trying to learn the parameters  $\beta$  and  $\delta$ . If the  $Q_1, \dots, Q_T$  are private information only available to the agent, we are in no better shape than in the case of  $\mathcal{P}_{\mathcal{W}}$ .



■ **Table 1** VC dimensions of various preference models (the lower bounds for  $\mathcal{P}_{\mathcal{PW}}$  and  $\mathcal{P}_{\mathcal{BPW}}$  require that  $Q_1, \dots, Q_T$  span the space of polynomials with degree at most  $T - 1$ ).

Preference Model $\mathcal{P}$	Payoff( $x$ )	$VC(\mathcal{P})$ lower bound	$VC(\mathcal{P})$ upper bound
$\mathcal{P}_{\mathcal{W}}$	$w \cdot x$ for $w \in \mathbb{R}^T$	$T - 1$	$T + 1$
$\mathcal{P}_{\mathcal{PW}}$ (defined with respect to polynomials $Q_1, \dots, Q_T$ )	$\sum_{t=1}^T Q_t(\delta)x_t$ for $\delta \in (0, 1)$ .	$\log(T - 1)$	$(1 + \varepsilon) \log d$ , for $\varepsilon > 0$ and $d = \max\{\deg Q_t\}$ large enough.
$\mathcal{P}_{\mathcal{BPW}}$ (defined with respect to polynomials $Q_1, \dots, Q_T$ )	$(\frac{1}{\beta} - 1) \sum_{t=1}^T Q_t(0)x_t$ $+ \sum_{t=1}^T Q_t(\delta)x_t$ for $\delta, \beta \in (0, 1)$ .	$\log(T - 1)$	$(1 + \varepsilon) \log d$ , for $\varepsilon > 0$ and $d = \max\{\deg Q_t\}$ large enough.
$\mathcal{P}_{\mathcal{D}}$	$\sum_{t=1}^T D(t)x_t$ for $D: [T] \rightarrow (0, 1)$ a decreasing map.	$T - 1$	$T + 1$
$\mathcal{P}_{\mathcal{ED}}$	$\sum_{t=1}^T \delta^t x_t$ for $\delta \in (0, 1)$ .	$\log(T - 1)$	$(1 + \varepsilon) \log(T - 1)$ for $\varepsilon > 0$ and $T$ large enough.
$\mathcal{P}_{\mathcal{QHD}}$	$x_1 + \beta \sum_{t=2}^T \delta^{t-1} x_t$ for $\beta, \delta \in (0, 1)$ .	$\log(T - 1)$	$(1 + \varepsilon) \log(T - 1)$ for $\varepsilon > 0$ and $T$ large enough.
$\mathcal{P}_{\mathcal{HD}}$	$\sum_{t=1}^T \frac{1}{1+t\alpha} x_t$ for $\alpha > 0$ .	$\log(T - 1)$	$(1 + \varepsilon) \log(T - 1)$ for $\varepsilon > 0$ and $T$ large enough.

We find that the stream-based model is in general difficult to analyze for the preference relations we work with. This difficulty seems to arise from the apparent need to quantify disagreement in order to explicitly write down learning guarantees. Though in most general situations it is unclear how to quantify disagreement for our preference relations, we present a redeeming situation for which we are able to provide a precise analysis of the learning guarantees for  $\mathcal{P}_{\mathcal{ED}}$ . Here, the analyst can learn  $\mathcal{P}_{\mathcal{ED}}$  with an exponential improvement in label complexity over the guarantees provided by the PAC model. This is achieved via a computation of the disagreement coefficient (defined in Section 5) of  $\mathcal{P}_{\mathcal{ED}}$  for a specific distribution.

► **Theorem 8.** *There exists a distribution  $\mu$  on  $\mathbb{R}^T \times \mathbb{R}^T$  for which the disagreement coefficient of  $\mathcal{P}_{\mathcal{ED}}$  is  $\theta = 2$ . Thus, for this distribution,*

$$\ell_{CAL}(\varepsilon) = \tilde{O}\left(\log T \log \frac{1}{\varepsilon}\right),$$

where the  $\tilde{O}$  notation suppresses terms that are logarithmic in  $\log T$  and  $\log 1/\varepsilon$ .

The measure  $\mu$  we construct is induced by the product Lebesgue measure on  $(0, 1)^{T-1}$ , and allows us to precisely translate statements about disagreement into statements about the roots of polynomials arising from a given choice. Once we have defined  $\mu$ , the calculation of  $\theta$  follows from basic probability arguments.

Now, in our case the analyst has structural information regarding the preference relation of the agent he is questioning. We find that allowing the analyst full control over the

membership queries he makes yields a learning algorithm that, despite its simplicity, takes advantage of this extra structure and yields a significant improvement in complexity over the stream-based setting. Additionally, the membership queries model naturally describes an experimental environment in which the analyst is able to ask the agent questions in real time.

We show that when the preference model satisfies some relatively benign structural requirements, even very naive algorithms outperform the guarantees provided by CAL in the stream-based setting (we are not aware of any lower bounds on the sample complexity of disagreement based active learning, so this only an improvement over the current sample complexity upper bound of the CAL algorithm). The example algorithm we give, relying on a simple binary search, has a query complexity of  $O(\log 1/\varepsilon)$ , which gets rid of the  $\log T$  dependence in Theorem 8.

The class of preference models is defined as follows: let  $g_1, \dots, g_T : \mathbb{R} \rightarrow \mathbb{R}$  be a collection of functions satisfying the properties listed in Section 5.3 and consider the model of preference relations  $\mathcal{P}$  parametrized by  $\delta$  where  $x \succsim y$  if and only if  $\sum_{t=1}^T g_t(\delta)x_t \geq \sum_{t=1}^T g_t(\delta)y_t$ <sup>7</sup>. We have (where  $M$  and  $C$  are constants that depend on  $g_1, \dots, g_T$ )

► **Proposition 9.** *There exists an algorithm that takes as input  $\varepsilon > 0$  and using  $O(\log \frac{MC}{\varepsilon})$  membership queries outputs  $\delta^h$  such that  $|\delta - \delta^h| \leq \varepsilon$ , where  $\delta$  parametrizes the target preference relation in  $\mathcal{P}$ .*

The remainder of the paper is devoted to proving the results discussed in this section.

## 4 PAC Learning

In this section we prove Theorems 5 and 6. We first note a preliminary upper bound due to Basu and Echenique [2]. Let  $\mathcal{P}_{\mathcal{I}}$  be the set of preference relations that satisfy the following axioms:

**Order:** For all  $x, y$  either  $x \succsim y$  or  $y \succsim x$  (completeness). For all  $x, y, z$ , if  $x \succsim y$  and  $y \succsim z$ , then  $x \succsim z$  (transitivity).

**Independence:** For all  $x, y, z$  and for any  $\lambda \in (0, 1)$ ,  $x \succsim y$  if and only if  $\lambda x + (1 - \lambda)z \succsim \lambda y + (1 - \lambda)z$ .

The class  $\mathcal{P}_{\mathcal{I}}$  satisfies the property that for any  $\succsim \in \mathcal{P}_{\mathcal{I}}$ , there are finitely many vectors  $q_1, \dots, q_K$ , with  $K \leq T$ , such that  $x \succsim y$  if and only if  $(q_k \cdot x)_{k=1}^K \geq_L (q_k \cdot y)_{k=1}^K$ , where  $\geq_L$  denotes the lexicographic order [5]. Then,  $\mathcal{P}_{\mathcal{PW}}, \mathcal{P}_{\mathcal{BPW}} \subset \mathcal{P}_{\mathcal{W}} \subset \mathcal{P}_{\mathcal{I}}$ , since the aforementioned characterization is satisfied with  $K = 1$  and  $q_1 = (w_1, \dots, w_T)$ .

This has two main consequences. First,  $\mathcal{P}_{\mathcal{PW}}$  and  $\mathcal{P}_{\mathcal{BPW}}$  (and thus all the discounted utility models) satisfy the measurability requirement discussed in Lemma 4 of [2] for the equivalence result of Theorem 2 to hold. Second, the VC dimensions of  $\mathcal{P}_{\mathcal{PW}}$  and  $\mathcal{P}_{\mathcal{BPW}}$  are all bounded above by  $T + 1$  (and in particular  $T - 1 \leq VC(\mathcal{P}_{\mathcal{W}}) \leq T + 1$ ). This follows due to Theorem 3.1 of [2], in which an argument similar to that required to compute the VC dimension of the class of half-spaces is used to show that  $VC(\mathcal{P}_{\mathcal{I}}) = T + 1$ . In all cases excluding the most general model of discounted utility, we are able to bring this down to  $\log(T - 1)$  (which we then show is tight by demonstrating the corresponding lower bound).

We begin by demonstrating that even in the discounted utility setting, without any structure we cannot do better than the learning bounds obtained for  $\mathcal{P}_{\mathcal{I}}$ .

<sup>7</sup> As before, the  $g_1, \dots, g_T$  are known to the analyst.

► **Proposition 4** (restated).  $T - 1 \leq VC(\mathcal{P}_{\mathcal{D}}) \leq T + 1$ .

**Proof.** That  $VC(\mathcal{P}_{\mathcal{D}}) \leq T + 1$  follows from Theorem 3.1 of [2], since  $\mathcal{P}_{\mathcal{D}} \subset \mathcal{P}_{\mathcal{I}}$ .

Here is a simple construction that shows  $VC(\mathcal{P}_{\mathcal{D}}) \geq T - 1$ . Fix an  $\varepsilon > 0$ . Let  $e_1, \dots, e_T$  be the standard unit vectors in  $\mathbb{R}^T$ , and consider the set of points  $\{(x^1, y^1), \dots, (x^{T-1}, y^{T-1})\}$ , where  $x^i = (1 - \varepsilon)e_i$  and  $y^i = e_{i+1}$ .

This set is shattered by  $\mathcal{P}_{\mathcal{D}}$ : for any  $(a_i)_{i=1}^{T-1}$ , choose  $D(1)$  arbitrarily from  $(0, 1)$ , and if  $D(i)$  has been defined, inductively define  $D(i + 1)$  such that  $D(i + 1) \leq D(i)(1 - \varepsilon)$  if  $a_i = 1$  and  $D(i) > D(i + 1) > (1 - \varepsilon)D(i)$  if  $a_i = 0$ . ◀

We now prove Theorems 5 and 6, which are restated below for convenience.

► **Theorem 5** (restated). *For every  $\varepsilon > 0$ , there exists a  $d_\varepsilon$  such that for every  $d \geq d_\varepsilon$  we have  $VC(\mathcal{P}_{\mathcal{PW}}), VC(\mathcal{P}_{\mathcal{BPW}}) \leq (1 + \varepsilon) \log d$  for any  $T$  and any  $T$  polynomials  $Q_1, \dots, Q_T$  of degree at most  $d$ .*

**Proof.** It suffices to establish the bound for  $\mathcal{P}_{\mathcal{BPW}}$ .

Let  $(z^1, \dots, z^n)$  be a set of points in  $\mathbb{R}^T \times \mathbb{R}^T$ ,  $z^i = (x^i, y^i)$ . For each  $z^i = (x^i, y^i)$ , define the plan  $f^i := x^i - y^i$ . Then, note that  $(z^1, \dots, z^n)$  is shattered by  $\mathcal{P}_{\mathcal{BPW}}$  if and only if  $((f^1, 0), \dots, (f^n, 0))$  is shattered by  $\mathcal{P}_{\mathcal{BPW}}$ . Hence, we may (and do) restrict attention to datasets of the form  $((f^1, 0), \dots, (f^n, 0))$ .

We have that  $((f^1, 0), \dots, (f^n, 0))$  is shattered by  $\mathcal{P}_{\mathcal{BPW}}$  if and only if for all vectors  $(a_1, \dots, a_n) \in \{0, 1\}^n$ , there exists a  $\delta$  and  $\beta$  (which determines the preference relation) such that

$$(Q_T(\delta)f_T^i + \dots + Q_1(\delta)f_1^i) + \left(\frac{1}{\beta} - 1\right) (Q_T(0)f_T^i + \dots + Q_1(0)f_1^i) \geq 0 \text{ whenever } a_i = 1,$$

and

$$(Q_T(\delta)f_T^i + \dots + Q_1(\delta)f_1^i) + \left(\frac{1}{\beta} - 1\right) (Q_T(0)f_T^i + \dots + Q_1(0)f_1^i) < 0 \text{ whenever } a_i = 0.$$

Therefore, if the  $n$  points  $((f^1, 0), \dots, (f^n, 0))$  can be shattered, there are polynomials  $P_1, \dots, P_n$  in  $\delta$  (where  $P_i$  is the polynomial  $Q_1(\delta)f_1^i + \dots + Q_T(\delta)f_T^i$ ), each of degree at most  $d$ , such that for every labeling  $(a_1, \dots, a_n) \in \{0, 1\}^n$ , there exists a  $\delta$  and  $\beta$  such that<sup>8</sup>

$$(\text{sgn}(P_1(\delta) + (1/\beta - 1)P_1(0)), \dots, \text{sgn}(P_n(\delta) + (1/\beta - 1)P_n(0))) = (a_1, \dots, a_n).$$

First, for any  $n$  polynomials  $P_1, \dots, P_n$  of degree at most  $d$ , we give an upper bound on the number of possible values  $(\text{sgn}(P_1(\delta)), \dots, \text{sgn}(P_n(\delta)))$  can realize. Each polynomial has at most  $d$  real roots, so together  $P_1, \dots, P_n$  have at most  $nd$  distinct real roots. Since sign changes can only occur at the roots, there are at most  $nd + 1$  possible values of  $\{0, 1\}^n$  that  $(\text{sgn}(P_1(\delta)), \dots, \text{sgn}(P_n(\delta)))$  can realize.

Now, for a fixed  $\delta$ , varying  $\beta$  shifts the collection of polynomials

$$P_1(\delta) + (1/\beta - 1)P_1(0), \dots, P_n(\delta) + (1/\beta - 1)P_n(0)$$

vertically, which in the worst case induces sign changes in all entries. We thus get at most an additional  $n$  new sign combinations for every sign combination realized by

<sup>8</sup> For notational convenience, let  $\text{sgn}(x)$  be 1 if  $x \geq 0$  and 0 otherwise.

## 62:12 Learning Time Dependent Choice

$(\text{sgn}(P_1(\delta)), \dots, \text{sgn}(P_n(\delta)))$ . Hence, there are at most  $nd + 1 + n(nd + 1) = (n^2 + n)d + n + 1$  possible values of  $\{0, 1\}^n$  that

$$(\text{sgn}(P_1(\delta) + (1/\beta - 1)P_1(0)), \dots, \text{sgn}(P_n(\delta) + (1/\beta - 1)P_n(0)))$$

can realize.

In order for all  $2^n$  elements of  $\{0, 1\}^n$  to be realized, it must be that

$$(n^2 + n)d + n + 1 \geq 2^n.$$

If  $n > (1 + \varepsilon) \log d$ , then for large enough  $d$  this inequality does not hold, and so any set of  $n$  points cannot be shattered. Thus, for all  $\varepsilon > 0$ ,  $n \leq (1 + \varepsilon) \log d$  for large enough  $d$ , so  $VC(\mathcal{P}_{\mathcal{B}\mathcal{W}}) \leq (1 + \varepsilon) \log d$ . ◀

We now establish the corresponding lower bound when the polynomials  $Q_1, \dots, Q_T$  span the space of polynomials of degree at most  $T - 1$ .

► **Theorem 6 (restated).** *Let  $Q_1, \dots, Q_T$  be polynomials in  $\delta$  of degree at most  $T - 1$  that span the space of polynomials in  $\delta$  of degree at most  $T - 1$ . Then  $VC(\mathcal{P}_{\mathcal{W}}), VC(\mathcal{B}\mathcal{P}\mathcal{W}) \geq \log(T - 1)$ .*

**Proof.** It suffices to establish the bound for  $\mathcal{P}_{\mathcal{W}}$ .

Consider the graph on  $\{0, 1\}^n$  where two vertices are connected by an edge if they differ in exactly one location. Fix a Hamiltonian path  $v_1, v_2, \dots, v_{2^n}$  in this graph (the existence of which is well known). Let  $b_{1,2}, \dots, b_{2^n-1,2^n}$  be the sequence where  $b_{i,i+1}$  is the index of the location at which  $v_i$  and  $v_{i+1}$  differ. Note that if  $n = \log(T - 1)$ , the graph has  $T - 1$  vertices, so each index in  $\{1, \dots, n\}$  can appear in the sequence  $(b_{i,i+1})$  at most  $T - 1$  times.

Now, let  $r_1 < r_2 < \dots < r_{2^n}$  be any points in  $(0, 1)$ . Define  $n$  polynomials  $P_1, \dots, P_n$  by  $P_k(\delta) = \prod_{b_{i,i+1}=k} (\delta - r_i)$ , so the roots of  $P_k$  are precisely the  $r_i$ 's that correspond to a flip in the entry at the  $k$ th position of a vertex in the path. Then,  $(\text{sgn}(P_1(\delta)), \dots, \text{sgn}(P_n(\delta)))$  realizes every element of  $\{0, 1\}^n$ .

Since  $Q_1, \dots, Q_T$  span the space of polynomials of degree at most  $T - 1$ , for each  $P_i$  we can find  $f_1^i, \dots, f_T^i$  such that

$$P_i(\delta) = Q_1(\delta)f_1^i + \dots + Q_T(\delta)f_T^i,$$

which gives us a collection of  $\log(T - 1)$  points that is shattered. Hence  $\log(T - 1) \leq VC(\mathcal{P}_{\mathcal{W}})$ . ◀

It is readily seen that  $\mathcal{P}_{\mathcal{E}\mathcal{D}}$  and  $\mathcal{P}_{\mathcal{Q}\mathcal{H}\mathcal{D}}$  satisfy the conditions of Theorems 5 and 6 ( $\mathcal{P}_{\mathcal{E}\mathcal{D}}$  is given by  $\mathcal{P}_{\mathcal{P}\mathcal{W}}$  with  $Q_t(\delta) = \delta^{t-1}$  and  $\mathcal{P}_{\mathcal{Q}\mathcal{H}\mathcal{D}}$  is given by  $\mathcal{B}\mathcal{P}\mathcal{W}$  with  $Q_t(\delta) = \delta^{t-1}$ ). We give a quick argument verifying that  $\mathcal{P}_{\mathcal{H}\mathcal{D}}$  does as well: For  $1 \leq t \leq T$ , let  $Q_t(\alpha) = \prod_{\ell \in \{1, \dots, T\} \setminus \{t\}} (1 + \ell\alpha)$  (these are the polynomials obtained by clearing denominators of the hyperbolic discount factors). We argue that  $\{Q_t(\alpha)\}_{t=1}^T$  are linearly independent over (and thus span) the vector space of polynomials in  $\alpha$  of degree at most  $T - 1$ . Indeed, if  $Q_1(\alpha)f_1 + \dots + Q_T(\alpha)f_T = 0$ , then we must have that  $f_1 = \dots = f_T = 0$  since at  $\alpha = \frac{-1}{t}$  we get  $Q_t(\alpha)f_t = 0$ . Hence,  $\mathcal{P}_{\mathcal{H}\mathcal{D}}$  is simply  $\mathcal{P}_{\mathcal{P}\mathcal{W}}$  with  $Q_t(\alpha) = \prod_{\ell \in \{1, \dots, T\} \setminus \{t\}} (1 + \ell\alpha)$ .

► **Remark.** These bounds also hold in the scenario where the agent can report indifference in the data. More precisely, the condition for  $x \succsim y$  is now a strict inequality, and we have three possible labels for the pair  $(x, y)$ :  $+1$  indicates that  $x \succ y$ ,  $-1$  indicates that  $y \succ x$ , and  $0$  indicates that  $x \sim y$ . Then, using a Hamiltonian path on  $\{-1, 0, 1\}^n$ , we can construct polynomials  $P_1, \dots, P_n$  such that  $(\text{sgn}(P_1(\delta)), \dots, \text{sgn}(P_n(\delta)))$  realizes all elements of  $\{-1, 0, 1\}^n$  as  $\delta$  ranges from 0 to 1 (where  $\text{sgn}$  is the true sign function).

## 4.1 A Remark on Efficient Learnability

While PAC learnability is a positive result, it does not take into account the computational complexity of computing a hypothesis. Blumer et. al. [6] show that any learning rule that outputs a hypothesis consistent with the data seen yields with high probability a hypothesis that has very low error. However, if the problem of outputting a consistent hypothesis is computationally intractable, PAC learnability on its own is perhaps unsatisfying. In this section we note that the discounted utility models of intertemporal choice are *efficiently* learnable. This is due to an algorithm of Grigor'ev and Vorobjov [12] for solving a system of polynomial inequalities.

For notational convenience, it will be useful to write  $\mathcal{P} = \{\mathcal{P}^T\}_{T \geq 1}$ , for each of the models above, where  $\mathcal{P}^T$  is the collection of preference relations for a given  $T$ . Moreover, suppose acts are chosen from  $[-1, 1]^T$  instead of  $\mathbb{R}^T$ . It is clear that this does not change any of the analysis above.

Polynomial learnability, as defined by Blumer et. al. [6], stipulates that the learning rule be computable in  $\text{poly}(1/\varepsilon, 1/\delta, T)$ -time (where  $\varepsilon$  and  $\delta$  denote the error threshold and confidence threshold respectively). Polynomial learnability is equivalent to the task of outputting a hypothesis consistent with the given data set in polynomial time [6].

► **Definition 10.** A *randomized polynomial hypothesis finder (r-poly hy-fi)* for  $\mathcal{P}$  is a randomized polynomial time algorithm that takes as input a sample of a preference relation in  $\mathcal{P}$ , and for some  $\gamma > 0$ , with probability at least  $\gamma$  produces a hypothesis that is consistent with the sample.

► **Theorem 11.**  $\mathcal{P}$  is properly polynomially learnable if and only if there is an r-poly hy-fi for  $\mathcal{P}$  and  $VC(\mathcal{P}^T)$  grows only polynomially in  $T$ .

We have just shown that  $VC(\mathcal{P}^T) \sim \log(T - 1)$ . Using techniques involving algebraic geometry, Grigor'ev and Vorobjov [12] present an algorithm to solve a system of  $N$  polynomial inequalities with a  $\text{poly}(N, T)$  runtime. This serves as our r-poly hy-fi, and thus we obtain:

► **Theorem 12.**  $\mathcal{P}_{\mathcal{E}\mathcal{D}}$ ,  $\mathcal{P}_{\mathcal{H}\mathcal{D}}$ , and  $\mathcal{P}_{\mathcal{Q}\mathcal{H}\mathcal{D}}$  are all properly polynomially learnable.

## 5 Active Learning

In this section we study two models of active learning: *stream-based selective sampling* and learning via *membership queries*. We first define a distribution for which the disagreement coefficient of  $\mathcal{P}_{\mathcal{E}\mathcal{D}}$  is 2, showing that disagreement methods (specifically the CAL algorithm [9, 10, 13]) in the stream-based model can yield an exponential improvement over the sample complexity of PAC learning (thus proving Theorem 8).

We then consider learning via membership queries and show that in this setting even very naive algorithms outperform the disagreement methods in the stream-based model (that is, the analyst needs to ask fewer questions to the agent in order to learn his preference than the number of label requests he would need to make using disagreement methods).

### 5.1 Preliminaries

For notational convenience,  $\succsim_\delta$  will refer to the preference relation in  $\mathcal{P}_{\mathcal{E}\mathcal{D}}$  with discounting factor  $\delta$ .

Let  $\mu$  be a distribution on  $\mathbb{R}^T \times \mathbb{R}^T$ .  $\mu$  induces a metric on  $\mathcal{P}_{\mathcal{E}\mathcal{D}}$  by  $d(\succsim_\delta, \succsim_\gamma) = \mu(\succsim_\delta \triangle \succsim_\gamma)$ , and thus we can define the closed ball of radius  $R$  centered at  $\succsim_\delta$  by

$$B(\succsim_\delta, R) = \{\succsim_\gamma: d(\succsim_\delta, \succsim_\gamma) \leq R\}.$$

## 62:14 Learning Time Dependent Choice

For  $V \subseteq \mathcal{P}_{\mathcal{E}\mathcal{U}}$ , the disagreement region of  $V$ ,  $\text{Dis}(V)$  is defined by

$$\text{Dis}(V) = \{(x, y) \in \mathbb{R}^T \times \mathbb{R}^T : \exists \succsim_\delta, \succsim_\gamma \in V \text{ s.t. } (x, y) \in \succsim_\delta \Delta \succsim_\gamma\} = \bigcup_{\succsim_\delta, \succsim_\gamma \in V} (\succsim_\delta \Delta \succsim_\gamma)$$

Intuitively,  $\text{Dis}(V)$  is the collection of points  $(x, y)$  such that we can find two hypothesis relations in the current version space that rank  $x$  and  $y$  differently.

If  $\succsim_\delta$  is the target preference relation, the disagreement coefficient of  $\succsim_\delta$  with respect to  $\mu$  is the quantity

$$\theta = \sup_{R>0} \frac{\mu(\text{Dis}(B(\succsim_\delta, R)))}{R}$$

### 5.2 Disagreement based active learning

In this subsection, we define a distribution  $\mu$  on  $\mathbb{R}^T \times \mathbb{R}^T$  and show that the disagreement coefficient of  $\mathcal{P}_{\mathcal{E}\mathcal{D}}$  with respect to  $\mu$  is 2.

#### Choosing a measure

The main challenge here is that  $\theta$  depends on the underlying distribution over  $\mathbb{R}^T \times \mathbb{R}^T$ . Since preferences are polynomial inequalities, the disagreement coefficient seems to lend itself to a characterization involving polynomials and their roots, which is the motivation for our choice of distribution. For a general distribution  $\mu$  over  $\mathbb{R}^T \times \mathbb{R}^T$ , it is not clear how to compute the disagreement coefficient.

We show that  $\theta = 2$  for a suitably chosen distribution on  $\mathbb{R}^T \times \mathbb{R}^T$ , which is induced by the Lebesgue measure on  $(0, 1)^{T-1}$ . This allows us to work with a measure on sets of roots of polynomials that arise from the definition of the preference relations.

Let  $\mu^{**}$  be a measure on  $(0, 1)^{T-1}$ . We interchangeably represent elements of  $\mathbb{R}^T$  as polynomials  $P$  of degree at most  $T - 1$  or as  $T - 1$ -tuples of coefficients. Let  $\sim$  be the equivalence relation on  $\mathbb{R}^T$  defined by  $P \sim Q \iff P = cQ$  for some constant  $c$ , and let  $\mathbb{R}^T / \sim$  be the resulting quotient space. Let  $g : (0, 1)^{T-1} \rightarrow \mathbb{R}^T / \sim$  be the map taking a tuple of roots to the equivalence class of the polynomials with those roots, and let  $h : \mathbb{R}^T \times \mathbb{R}^T \rightarrow \mathbb{R}^T / \sim$  be the map  $h(x, y) = [x - y]$ .

We define the following measures  $\mu^*$  and  $\mu$  on  $g((0, 1)^{T-1})$  and  $h^{-1}(g((0, 1)^{T-1}))$  respectively.

- Define  $\mu^*$  on all sets  $S \subset g((0, 1)^{T-1})$  such that

$$\{(r_1(P), \dots, r_{T-1}(P)) \in (0, 1)^{T-1} : P \in S\}$$

(where  $r_1(P), \dots, r_{T-1}(P)$  denote the roots of  $P$ ) is  $\mu^{**}$ -measurable, for which we set

$$\mu^*(S) = \mu^{**}(\{(r_1(P), \dots, r_{T-1}(P)) \in (0, 1)^{T-1} : P \in S\}).$$

- Define  $\mu$  on all sets  $S \subset h^{-1}(g((0, 1)^{T-1}))$  such that

$$\{[z] \in g((0, 1)^{T-1}) : \exists(x, y) \in S \text{ s.t. } z \sim x - y\}$$

is  $\mu^*$ -measurable, for which we set

$$\mu(S) = \mu^*(\{[z] \in g((0, 1)^{T-1}) : \exists(x, y) \in S \text{ s.t. } z \sim x - y\}).$$

Intuitively,  $\mu^*$  is defined only on those polynomials that have all their roots in  $(0, 1)$ . When  $T = 2$ , this is a desirable property since the analyst is only presented with polynomials that have some disagreement in  $(0, 1)$ . He is not presented with meaningless polynomials that are, for example, always positive on  $(0, 1)$  (the analyst has nothing to learn from such polynomials since such a polynomial will be preferred to 0 for all  $\delta \in (0, 1)$ ). For  $T \geq 3$ , this is a more restrictive property since the analyst is only presented with polynomials that have all  $T - 1$  roots in  $(0, 1)$ .

Let  $\mu^{**}$  be the product Lebesgue measure on  $(0, 1)^{T-1}$ . Choosing  $\mu^{**}$  in this fashion allows us to neatly characterize  $B(\succsim_\delta, R)$ .

Let  $X_1, \dots, X_{T-1}$  be uniform i.i.d. random variables on  $(0, 1)$ , and let  $Y_{\delta, \gamma}$  be the random variable  $Y_{\delta, \gamma} = |\{i : X_i \text{ is between } \delta \text{ and } \gamma\}|$ . Let  $E_{\delta, \gamma}^{odd}$  denote the event that  $Y_{\delta, \gamma}$  is odd, let  $E_{\delta, \gamma}^k$  denote the event  $Y_{\delta, \gamma} = k$ , and let  $E_{\delta, \gamma}^{\geq k}$  denote the event  $Y_{\delta, \gamma} \geq k$ .

► **Lemma 13.**  $\succsim_\gamma \in B(\succsim_\delta, R)$  if and only if  $\mathbb{P}[E_{\delta, \gamma}^{odd}] \leq R$ .

**Proof.** Given  $(x, y) \in \mathbb{R}^T \times \mathbb{R}^T$ , let  $P_{x-y}(X) = \sum_{t=1}^T X^{t-1} \cdot (x_t - y_t)$ . Then,

$$\begin{aligned} \succsim_\gamma \in B(\succsim_\delta, R) & \\ \iff \mu(\{(x, y) \in h^{-1}(g((0, 1)^{T-1})) : \text{sgn}(P_{x-y}(\delta)) \neq \text{sgn}(P_{x-y}(\gamma))\}) & \leq R \\ \iff \mu^*(\{[P] \in g((0, 1)^{T-1}) : \text{sgn}(P(\delta)) \neq \text{sgn}(P(\gamma))\}) & \leq R \\ \iff \mu^{**}(\{(r_1, \dots, r_{T-1}) \in (0, 1)^{T-1} : \text{sgn}(\prod(\delta - r_i)) \neq \text{sgn}(\prod(\gamma - r_i))\}) & \leq R \end{aligned}$$

But  $\text{sgn}(\prod(\delta - r_i)) \neq \text{sgn}(\prod(\gamma - r_i))$  occurs exactly when an odd number of roots lie between  $\gamma$  and  $\delta$  (modulo a set of measure 0 since the probability that we have a root of multiplicity greater than 1 is 0). ◀

While it seems difficult to write down a general characterization of  $\mu$ , Propositions 14 and 15 give some basic observations regarding the  $\sigma$ -algebras on which  $\mu^*$  and  $\mu$  are defined. We defer their proofs (along with a description of  $\mu^*$  in the case  $T = 2$ ) to the appendix:

► **Proposition 14.** The  $\sigma$ -algebra on  $g((0, 1)^{T-1})$  is the Borel  $\sigma$ -algebra.

The  $\sigma$ -algebra induced on  $h^{-1}(g((0, 1)^{T-1}))$  does not appear to yield a clean characterization, but we can show the weaker statement that  $\mu$  is a Borel measure, i.e. it is defined on all open sets of  $h^{-1}(g((0, 1)^{T-1}))$ .

► **Proposition 15.**  $\mu$  is a Borel measure on  $h^{-1}(g((0, 1)^{T-1}))$ .

## Computing $\theta$

We now show  $\theta = 2$  for the distribution  $\mu$  as chosen above. We use the notation  $d = d_{\delta, \gamma} := |\delta - \gamma|$  to denote the distance between  $\delta$  and  $\gamma$ . Let  $\succsim_\delta$  be the target preference relation.

First, note that since  $Y_{\delta, \gamma}$  is distributed according to  $\text{Bin}(T-1, d)$ ,  $\mathbb{P}[E_{\delta, \gamma}^{odd}] = \frac{1 - (1-2d)^{T-1}}{2}$ <sup>9</sup>. We use this fact to derive an explicit description of the preference relations  $\succsim_\gamma$  contained in the ball  $B(\succsim_\delta, R)$  in terms of  $d$ . We break the analysis up into a few cases.

First, when  $R > \frac{1}{2}$ , we have  $\sup_{R > \frac{1}{2}} \frac{\mu(\text{Dis}(B(\succsim_\delta, R)))}{R} = 2$ . Let  $R \leq \frac{1}{2}$ . By Lemma 13,

$$\succsim_\gamma \in B(\succsim_\delta, R) \iff \mathbb{P}[E_{\delta, \gamma}^{odd}] = \frac{1 - (1 - 2d)^{T-1}}{2} \leq R \quad (1)$$

<sup>9</sup> This is due to the general fact that if  $X$  is a random variable distributed according to  $\text{Bin}(n, p)$ , the probability that  $X$  is odd is  $\frac{1 - (1-2p)^n}{2}$ .



## 62:16 Learning Time Dependent Choice

Suppose  $d \leq \frac{1}{2}$ . Then  $1 - 2d$  and  $1 - 2R$  are both non-negative, so rearranging Equation (1) yields

$$d \leq \frac{1 - (1 - 2R)^{1/(T-1)}}{2}. \quad (2)$$

Suppose  $d > \frac{1}{2}$ , so  $1 - 2d < 0$ . Rearranging Equation (1), we get  $1 - 2R \leq (1 - 2d)^{T-1}$ . If  $T - 1$  is odd,  $(1 - 2d)^{T-1}$  is negative, so  $1 - 2R \leq (1 - 2d)^{T-1}$  does not hold. Thus, when  $T - 1$  is odd the ball consists of  $\zeta_\gamma$  such that  $d$  satisfies condition (2). If  $T - 1$  is even,  $(1 - 2d)^{T-1}$  is positive, so we get

$$d \geq \frac{1 + (1 - 2R)^{1/(T-1)}}{2}. \quad (3)$$

Thus, when  $T - 1$  is even the ball consists of  $\zeta_\gamma$  such that  $d$  satisfies conditions (2) or (3).

Now, the disagreement region of  $B(\zeta_\delta, R)$  consists of all points  $(x, y)$  such that the polynomial  $P_{x-y}$  (as defined in Lemma 13) has a root  $\gamma$  such that  $\zeta_\gamma \in B(\zeta_\delta, R)$  (since we can find two hypotheses that disagree on  $(x, y)$  by taking a point slightly below  $\gamma$  and a point slightly above  $\gamma$  such that  $P_{x-y}$  has no sign changes in between). Hence, with  $R_1 = \frac{1 - (1 - 2R)^{1/(T-1)}}{2}$  and  $R_2 = \frac{1 + (1 - 2R)^{1/(T-1)}}{2}$ , we have that

$$\mu(\text{Dis}(B(\zeta_\delta, R))) = \begin{cases} \mathbb{P}[E_{\delta-R_1, \delta+R_1}^{\geq 1}] & \text{if } T - 1 \text{ is odd} \\ \mathbb{P}[E_{\delta-R_1, \delta+R_1}^{\geq 1} \cup E_{0, \delta-R_2}^{\geq 1} \cup E_{\delta+R_2, 1}^{\geq 1}] & \text{if } T - 1 \text{ is even} \end{cases}$$

We have

$$\mathbb{P}[E_{\delta-R_1, \delta+R_1}^{\geq 1}] = 1 - (1 - 2R_1)^{T-1} = 2R,$$

and

$$\mathbb{P}[E_{\delta-R_1, \delta+R_1}^{\geq 1} \cup E_{0, \delta-R_2}^{\geq 1} \cup E_{\delta+R_2, 1}^{\geq 1}] = 1 - (2(R_2 - R_1))^{T-1} = 1 - 2^{T-1}(1 - 2R).$$

Therefore, when  $T - 1$  is odd  $\sup_{0 < R \leq 1/2} \frac{\mu(\text{Dis}(B(\zeta_\delta, R)))}{R} = 2$  and when  $T - 1$  is even

$$\sup_{0 < R \leq 1/2} \frac{\mu(\text{Dis}(B(\zeta_\delta, R)))}{R} = \sup_{0 < R \leq 1/2} \frac{1 - 2^{T-1}(1 - 2R)}{R} = 2,$$

which is achieved at  $R = 1/2$  since  $\frac{1 - 2^{T-1}(1 - 2R)}{R}$  is increasing on  $0 < R \leq \frac{1}{2}$ .

Finally,  $\theta = \sup_{R > 0} \frac{\mu(\text{Dis}(B(\zeta_\delta, R)))}{R} = 2$ .

We have thus established Theorem 8:

► **Theorem 8 (restated).** *There exists a distribution  $\mu$  on  $\mathbb{R}^T \times \mathbb{R}^T$  for which the disagreement coefficient of  $\mathcal{P}_{\mathcal{E}\mathcal{D}}$  is  $\theta = 2$ . Thus, for this distribution,*

$$\ell_{CAL}(\varepsilon) = \tilde{O}\left(\log T \log \frac{1}{\varepsilon}\right),$$

where the  $\tilde{O}$  notation suppresses terms that are logarithmic in  $\log T$  and  $\log 1/\varepsilon$ .

### 5.3 Learning via membership queries

In this subsection, we present a simple membership queries algorithm that outperforms the guarantees provided by the disagreement based CAL algorithm. For simplicity, we restrict attention to preference models that are parametrized by a single parameter (e.g.  $\mathcal{P}_{\mathcal{E}\mathcal{D}}$  and  $\mathcal{P}_{\mathcal{H}\mathcal{D}}$ ).

Let  $g_1, \dots, g_T : \mathbb{R} \rightarrow \mathbb{R}$  be a collection of functions such that there exist  $1 \leq t_1, t_2 \leq T$  satisfying

1.  $M := \sup_{\delta} \frac{g_{t_1}(\delta)}{g_{t_2}(\delta)}$  is finite, and
2. The map  $\delta \mapsto \frac{g_{t_1}(\delta)}{g_{t_2}(\delta)}$  satisfies an inverse Lipschitz condition with constant  $C$  :

$$|\delta - \delta'| \leq C \left| \frac{g_{t_1}(\delta)}{g_{t_2}(\delta)} - \frac{g_{t_1}(\delta')}{g_{t_2}(\delta')} \right|.$$

Consider the model of preference relations  $\mathcal{P}$  parametrized by  $\delta$  such that

$$x \succsim y \text{ if and only if } \sum_{t=1}^T g_t(\delta)x_t \geq \sum_{t=1}^T g_t(\delta)y_t.$$

► **Proposition 9 (restated).** *There exists an algorithm that takes as input  $\varepsilon > 0$  and using  $O(\log \frac{MC}{\varepsilon})$  membership queries outputs  $\delta^h$  such that  $|\delta - \delta^h| \leq \varepsilon$ , where  $\delta$  parametrizes the target preference relation in  $\mathcal{P}$ .*

**Proof.** Fix a  $\rho > 0$  and an  $\eta$ -cover of  $[0, M\rho]$ , where  $0 < \eta \leq \frac{\rho\varepsilon}{C}$ . Let  $b_\rho$  be the quantity such that the agent is indifferent between receiving a payoff of  $\rho$  at time  $t_1$  or receiving a payoff of  $b_\rho$  at time  $t_2$ , i.e.  $b_\rho$  solves

$$g_{t_2}(\delta)b_\rho = g_{t_1}(\delta)\rho.$$

By running a binary search over the  $\eta$ -cover of  $[0, M\rho]$ , the analyst can find an approximation  $b_\rho^h$  to the indifference point for which  $|b_\rho - b_\rho^h| \leq \eta$  (the binary search is performed on the parameter  $b_\rho^h$  by requesting labels for pairs of the form  $(\rho e_{t_1}, b_\rho^h e_{t_2})$ ). The analyst then outputs the  $\delta^h$  that solves  $g_{t_2}(\delta^h)b_\rho^h = g_{t_1}(\delta^h)\rho$ .

We have

$$|\delta - \delta^h| \leq C \left| \frac{g_{t_1}(\delta)}{g_{t_2}(\delta)} - \frac{g_{t_1}(\delta^h)}{g_{t_2}(\delta^h)} \right| = C \left| \frac{b_\rho}{\rho} - \frac{b_\rho^h}{\rho} \right| \leq \frac{C\eta}{\rho} \leq \varepsilon,$$

as desired.

Since  $M := \sup_{\delta} \frac{g_{t_1}(\delta)}{g_{t_2}(\delta)}$  is finite,  $b_\rho \leq M\rho$ , so the binary search over the  $\eta$ -cover of  $[0, M\rho]$  terminates. ◀

► **Remark.** Outputting a hypothesis parameter  $\delta^h$  that is  $\varepsilon$ -close to  $\delta$  is a reasonable measurement for the error of learning via membership queries since there is no underlying distribution providing points to the analyst. However, note that for a distribution on  $\mathbb{R}^T \times \mathbb{R}^T$ , a hypothesis close to the target parameter implies the set of misclassified points is assigned a small measure, due to continuity of measure.

The main feature of this algorithm is that its query complexity has no dependence on the number of time periods  $T$ . Both  $\mathcal{P}_{\mathcal{E}\mathcal{D}}$  and  $\mathcal{P}_{\mathcal{H}\mathcal{D}}$  fit the conditions of Proposition 9, and thus we obtain a large improvement over the guarantees provided by disagreement methods in the stream-based model. Such methods assume no extra knowledge about the problem domain and are written to fit a wide class of learning problems. When we are learning economic parameters, membership queries allow us to take advantage of the extra structure present in preference models.

## References

- 1 M.F. Balcan, A. Daniely, R. Mehta, R. Urner, and V. V. Vazirani. Learning economic parameters from revealed preferences. In *International Conference on Web and Internet Economics*, pages 338–353. Springer, Cham, 2014.
- 2 P. Basu and F. Echenique. Learnability and Models of Decision Making under Uncertainty. In *Proceedings of the 2018 ACM Conference on Economics and Computation*, pages 53–53. ACM, 2018.
- 3 E. Beigman and R. Vohra. Learning from revealed preference. In *Proceedings of the 7th ACM Conference on Electronic Commerce*, pages 36–42. ACM, 2006.
- 4 G. S. Berns, D. Laibson, and G. Loewenstein. Intertemporal choice – toward an integrative framework. *Trends in cognitive sciences*, 11(11):482–488, 2006.
- 5 L. Blume, A. Brandenburger, and E. Dekel. Lexicographic probabilities and choice under uncertainty. *Econometrica: Journal of the Econometric Society*, pages 61–79, 1991.
- 6 A. Blumer, A. Ehrenfeucht, D. Haussler, and M. K. Warmuth. Learnability and the Vapnik-Chervonenkis dimension. *Journal of the ACM (JACM)*, 36(4):929–965, 1989.
- 7 C. F. Chabris, D. I. Laibson, and J. P. Schuldt. Intertemporal choice. *Behavioural and Experimental Economics*, pages 168–177, 2010.
- 8 C. P. Chambers and F. Echenique. On multiple discount rates. *Econometrica*, 86(4):1325–1346, 2018.
- 9 D. Cohn, L. Atlas, and R. Ladner. Improving generalization with active learning. *Machine learning*, 15(2):201–221, 1994.
- 10 S. Dasgupta. Two faces of active learning. *Theoretical computer science*, 412(19):1767–1781, 2011.
- 11 F. Echenique, D. Golovin, and A. Wierman. A revealed preference approach to computational complexity in economics. In *Proceedings of the 12th ACM conference on Electronic commerce*, pages 929–965. ACM, 1989.
- 12 D. Grigoriev and N. Vorobjov. Solving systems of polynomial inequalities in subexponential time. *J. Symb. Comput.*, 5(1/2):37–64, 1988.
- 13 S. Hanneke. Theoretical foundations of active learning. *CARNEGIE-MELLON UNIV PITTSBURGH PA MACHINE LEARNING DEPT.*, 2009.
- 14 S. Hanneke. The optimal sample complexity of PAC learning. *The Journal of Machine Learning Research*, 17(1):1319–1333, 2016.
- 15 J. W. Kable and P. W. Glimcher. The neural correlates of subjective value during intertemporal choice. *Nature neuroscience*, 10(12):1625, 2007.
- 16 G. Kalai. Learnability and rationality of choice. *Journal of Economic theory*, 113(1):104–117, 2003.
- 17 J. Kleinberg and S. Oren. Time-inconsistent planning: a computational problem in behavioral economics. In *Proceedings of the fifteenth ACM conference on Economics and computation*, pages 547–564. ACM, 2014.
- 18 T. C. Koopmans. Stationary ordinal utility and impatience. *Econometrica: Journal of the Econometric Society*, pages 287–309, 1960.
- 19 E. S. Phelps and R. A. Pollak. On second-best national saving and game-equilibrium growth. *The Review of Economic Studies*, 35(2):185–199, 1968.
- 20 N. Stern, S. Peters, V. Bakhshi, A. Bowen, C. Cameron, S. Catovsky, ..., and N. Edmonson. *Stern Review: The economics of climate change*. London: HM treasury, 2006.
- 21 M. Zadimoghaddam and A. Roth. Efficiently learning from revealed preference. In *International Workshop on Internet and Network Economics*, pages 114–127. Springer, Berlin, Heidelberg, 2012.

### A Properties of $\mu$ and $\mu^*$

**Proof of Proposition 14.** Let  $\mathcal{B}$  denote the Borel  $\sigma$ -algebra on  $(0, 1)^{T-1}$ .

Let  $X = \mathbb{R}^T / \sim$  endowed with the quotient topology, and  $g^* : [0, 1]^{T-1} \rightarrow X$  be the map  $g^*(r_1, \dots, r_{T-1}) = [\prod (x - r_i)]$ . Explicitly, the terms of  $g^*(r_1, \dots, r_{T-1})$  are given by symmetric sums:

$$g^*(r_1, \dots, r_{T-1}) = \left( c, -c \sum_i r_i, c \sum_{i,j} r_i r_j, \dots, (-1)^{T-1} c r_1 \cdots r_{T-1} \right),$$

where  $c$  is the appropriate constant for the representative of the equivalence class. Each symmetric sum is a continuous function of  $T - 1$  variables, so  $g^*$  is continuous. Moreover, note that  $g^*$  is injective. Then, with  $Y = g^*([0, 1]^{T-1})$ , we have that  $g^* : [0, 1]^{T-1} \rightarrow Y$  is a continuous bijection from a compact set into a Hausdorff space. Hence,  $g^*$  is a homeomorphism. Then  $g$ , which is the restriction of  $g^*$  to  $(0, 1)^{T-1}$  is a homeomorphism onto  $Z := g((0, 1)^{T-1})$ . Thus, the  $\sigma$ -algebra  $g(\mathcal{B})$  that we obtain on  $Z$  is the Borel  $\sigma$ -algebra.  $\blacktriangleleft$

When  $T = 2$ , we can give an explicit description of  $\mu^*$ . Identify  $\mathbb{R}^2 / \sim$  with the unit circle. Then, a degree 1 polynomial  $P$  is identified with the point  $(\cos \theta, \sin \theta)$ , where  $P(x) = (\cos \theta)x + \sin \theta$ .  $Z := g((0, 1)^{T-1})$  consists of the boundary of the unit circle for which the argument  $\theta$  satisfies  $-\tan \theta \in (0, 1)$ . This is satisfied precisely for  $\theta \in (3\pi/4, \pi) \cup (7\pi/4, 2\pi)$ . Hence, if  $U$  is a basic open subset of  $\{(\cos \theta, \sin \theta) : \theta \in (3\pi/4, \pi) \cup (7\pi/4, 2\pi)\}$ , we can write  $U = \{(\cos \theta, \sin \theta) : \theta_1 < \theta < \theta_2\}$  with  $\theta_1, \theta_2$  both in the same segment of the unit circle and

$$\mu^*(U) = \mu^{**}(\{-\tan \theta : \theta_1 < \theta < \theta_2\}) = |\tan \theta_1 - \tan \theta_2|.$$

**Proof of Proposition 15.** Let  $h : \mathbb{R}^T \times \mathbb{R}^T \rightarrow \mathbb{R}^T / \sim$  be the map  $h(x, y) = [x - y]$ . Let  $V \subseteq h^{-1}(g((0, 1)^{T-1}))$  be open. We show that

$$h(V) = \{[z] : \exists (x, y) \in V (z = x - y)\}$$

is open. Indeed, let  $z = x - y$  for  $(x, y) \in V$  and choose  $\varepsilon$  small enough such that the square with vertices  $\{(x + \varepsilon, y + \varepsilon), (x + \varepsilon, y - \varepsilon), (x - \varepsilon, y + \varepsilon), (x - \varepsilon, y - \varepsilon)\}$  is contained in  $V$ . Then, for any  $\lambda \leq \varepsilon$ ,  $[z + \lambda] = [(x + \lambda) - y]$  with  $(x + \lambda, y) \in V$  and  $[z - \lambda] = [x - (y + \lambda)]$  with  $(x, y + \lambda) \in V$ , so in particular the open ball with radius  $\lambda$  centered at  $[z]$  is contained in  $h(V)$ .  $\blacktriangleleft$



# Erasures vs. Errors in Local Decoding and Property Testing

**Sofya Raskhodnikova**


Department of Computer Science, Boston University, USA  
sofya@bu.edu

**Noga Ron-Zewi**

Department of Computer Science, University of Haifa, Israel  
noga@cs.haifa.il

**Nithin Varma**

Department of Computer Science, Boston University, USA  
nvarma@bu.edu

 <https://orcid.org/0000-0002-1211-2566>

---

## Abstract

---

We initiate the study of the role of erasures in local decoding and use our understanding to prove a separation between erasure-resilient and tolerant property testing. Local decoding in the presence of errors has been extensively studied, but has not been considered explicitly in the presence of erasures.

Motivated by applications in property testing, we begin our investigation with local *list* decoding in the presence of erasures. We prove an analog of a famous result of Goldreich and Levin on local list decodability of the Hadamard code. Specifically, we show that the Hadamard code is locally list decodable in the presence of a constant fraction of erasures, arbitrary close to 1, with list sizes and query complexity better than in the Goldreich-Levin theorem. We use this result to exhibit a property which is testable with a number of queries independent of the length of the input in the presence of erasures, but requires a number of queries that depends on the input length,  $n$ , for tolerant testing. We further study *approximate* locally list decodable codes that work against erasures and use them to strengthen our separation by constructing a property which is testable with a constant number of queries in the presence of erasures, but requires  $n^{\Omega(1)}$  queries for tolerant testing.

Next, we study the general relationship between local decoding in the presence of errors and in the presence of erasures. We observe that every locally (uniquely or list) decodable code that works in the presence of errors also works in the presence of twice as many erasures (with the same parameters up to constant factors). We show that there is also an implication in the other direction for locally decodable codes (with unique decoding): specifically, that the existence of a locally decodable code that works in the presence of erasures implies the existence of a locally decodable code that works in the presence of errors and has related parameters. However, it remains open whether there is an implication in the other direction for locally *list* decodable codes<sup>1</sup>. We relate this question to other open questions in local decoding.

**2012 ACM Subject Classification** Theory of computation → Streaming, sublinear and near linear time algorithms, Mathematics of computing → Coding theory

**Keywords and phrases** Error-correcting codes, probabilistically checkable proofs (PCPs) of proximity, Hadamard code, local list decoding, tolerant testing

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2019.63

---

<sup>1</sup> Our Hadamard result shows that there has to be some difference in parameters for some settings.



**Related Version** A full version [36] (<https://eccc.weizmann.ac.il/report/2018/195>) of the paper contains all the omitted proofs.

**Funding** The first and the third author were supported by National Science Foundation under Grant No. CCF-142297.

**Acknowledgements** The authors are thankful to Venkatesan Guruswami for helping to tighten the analysis of the local list erasure-decoder for the Hadamard code and also for making a suggestion that led to Observation 4.2. The authors are grateful to Prahladh Harsha, Or Meir, Ramesh Krishnan S. Pallavoor, Adam Smith, Sergey Yekhanin, and Avi Wigderson for useful discussions. Last but not least, the authors express their gratitude to the sponsors and organizers of the Workshop on Local Algorithms 2018 for making this collaboration possible.

## 1 Introduction

The contributions of this work are two-fold: on one hand, we initiate the investigation of erasures in local decoding; on the other hand, we apply our understanding of local list decoding to study the relative difficulty with which sublinear algorithms can cope with erasures and errors in their inputs.

Intuitively, a family of codes is *locally decodable* in the presence of a specified type of corruptions (erasures or errors) if there exists an algorithm that, given oracle access to a codeword with a limited fraction of specified corruptions, can decode each desired character of the encoded message with high probability after querying a small number of characters in the corrupted codeword. In other words, we can simulate oracle access to the message by using oracle access to a corrupted codeword. This notion can be extended to local *list* decoding by requiring the algorithm to output a list of descriptions of local decoders. Intuitively, a family of codes is *locally list decodable* in the presence of a specified type of corruptions if there exists an algorithm that, given oracle access to a corrupted codeword  $w$ , outputs a list of algorithms such that for each message  $x$  whose encoding sufficiently agrees with  $w$ , there is an algorithm in the list that, given oracle access to  $w$ , can simulate oracle access to  $x$ . In addition to the usual quantities studied in the literature on error-correcting codes (such as the fraction of corruptions a code can handle, its rate and efficiency of decoding), the important parameters in local decoding are the number of queries that the algorithms make to  $w$  and, in the case of local list decoding, list size.

The notion of locally decodable codes (LDCs) arose in the 1990s, motivated by numerous applications in complexity theory, such as program checking, probabilistically checkable proofs, derandomization, and private information retrieval. Locally decodable codes that work in the presence of errors have been extensively studied [2, 9, 16, 17, 35, 3, 40, 14, 13, 5]. The related notion of locally list decodable codes (LLDCs) has also received a lot of attention [19, 38, 27, 5, 31, 29, 22, 20] and found applications in cryptography, learning theory, average-to-worst-case reductions, and hardness amplification and derandomization. The literature on decoding in the presence of erasures is too vast to survey here. *List* decoding in the presence of erasures (without the locality restriction) has been addressed by Guruswami [23] and Guruswami and Indyk [24]. In particular, Guruswami [23] constructed an asymptotically good family of binary linear codes that can be list decoded from an arbitrary fraction of erasures with lists of constant size. Even though decoding in the presence of erasures is an important and well established problem, to the best of our knowledge, local (unique and list) decoding from erasures has not been studied before.



Motivated by applications in property testing [18, 37], we begin our investigation of effects of erasures with local *list* decoding. Our first result is a local list *erasure-decoder* for the Hadamard code. Local list decodability of the Hadamard code in the presence of errors is a famous result of Goldreich and Levin [19]. However, (local list) decoding of the Hadamard code is impossible when the fraction of errors reaches or exceeds  $1/2$ . In contrast, we show that the Hadamard code is locally list decodable in the presence of any constant fraction of erasures in  $[0, 1)$ . Moreover, the list size and the query complexity for our decoder is better than for the Goldreich-Levin decoder: for our decoder, both quantities are inversely proportional to the fraction of input that has not been corrupted, whereas for the Goldreich-Levin decoder they are quadratically larger and are known to be optimal for that setting. Thus, our Hadamard decoder demonstrates that a square-root reduction in the list size and query complexity in local list decoding can be achieved for some settings of parameters when we move from errors to erasures.

The second thrust of our work, enabled by our local list decoding results, is investigating the effects of adversarial corruption to inputs on the complexity of sublinear-time algorithms. Understanding the relative difficulty of designing algorithms that work in the presence of input errors and in the presence of input erasures is a problem of fundamental importance. The motivation of investigating adversarial input corruption spurred the generalization of property testing, one of the most widely studied models of sublinear-time algorithms, to (error) tolerant testing [34] and erasure-resilient testing [12].

Erasure-resilient property testing falls between (standard) property testing and tolerant testing. Specifically, an erasure-resilient tester for a property, in the special case when no erasures occur, is a standard tester for this property. Also, a tolerant tester for a property implies the existence of an erasure-resilient tester with comparable parameters for the same property. Fischer and Fortnow [15] separated standard and tolerant testing by describing a property that is *easy* to test in the standard model and *hard* to test tolerantly. Dixit et al. [12] showed that the property defined by Fischer and Fortnow separates standard property testing from erasure-resilient testing in the same sense. Dixit et al. [12] asked whether it is possible to obtain a separation between erasure-resilient and tolerant testing.

In this work, we provide such a separation. Specifically, we describe a property of binary strings that is easy to test in the erasure-resilient model, but hard to test tolerantly.

The key idea in our construction of the separating property is to encode *sensitive regions* of strings (without which testing becomes hard) with an error correcting code. We need a code that exhibits a difference in its local list decoding capabilities for the same fraction of erasures and errors. Specifically, we want, for some constant  $\alpha, q$  and  $L$ , a code that can be decoded from an  $\alpha$  fraction of erasures with  $q$  queries and lists of size  $L$ , but cannot be decoded from an  $\alpha$  fraction of errors. We first define a property where the sensitive regions are encoded with the Hadamard code and show that it is testable in the erasure-resilient model (with a constant number of queries), but is not testable tolerantly.

Next, we want to strengthen the separation to obtain a property that is testable with erasures, but requires as many queries as possible to test tolerantly. In our construction, the lower bound on the number of queries needed for tolerant testing is determined by the rate of the code. Since the Hadamard code has low rate, we only get a polylogarithmic lower bound on the query complexity of tolerant testing. To obtain a lower bound of  $n^{\Omega(1)}$ , we would need a code of polynomial rate. The question of whether there is a locally list erasure decodable code (with constant  $\alpha, q$  and  $L$ ) of polynomial rate remains open. An LLDC with such parameters is the holy grail of research on local decoding.

We circumvent the above difficulty by starting out with a property of binary strings that has a tester whose queries to a sensitive region of the input are *nearly uniformly* distributed. This implies that testing remains easy even if a constant fraction of the sensitive region is

corrupted. We construct a new separating property by encoding the sensitive region using a code that is *approximate locally list decodable* from erasures, where an approximate locally list decodable code (ALLDC) is defined identically to an LLDC except that the algorithms output by a decoder for such a code simulate oracle access to strings that are close to the original messages. We show that the resulting property can be erasure-resiliently tested using a constant number of queries but needs  $n^{\Omega(1)}$  queries in order to be tested tolerantly, thus obtaining a strengthened separation.

Next, we study the general relationship between local decoding in the presence of errors and in the presence of erasures. One can observe that every LLDC that works in the presence of errors also works in the presence of twice as many erasures (with the same parameters up to constant factors). We ask if LLDCs or ALLDCs that work in the presence of erasures can have significantly smaller list sizes and query complexity than LLDCs or ALLDCs of the same rate that work in the presence of errors. We also prove that such a statement cannot hold for the case of local unique decoding: specifically, we show that if a code is locally unique erasure-decodable, then there exists another comparable code that is locally unique decodable (up to minor losses in parameters).

## 1.1 Model Definitions and Our Results

This section contains descriptions and definitions of the codes and property testing models we study, and also statements and discussion of our main results.

### Local List Erasure-Decoding and the Hadamard Code

In this paper, we restrict our attention to binary codes. A binary code is an infinite family of maps  $\{C_n : \mathbb{F}_2^n \rightarrow \mathbb{F}_2^N\}_{n \in \mathbb{N}}$ . The parameter  $n$  is called the message length,  $N$  is the block length, and  $n/N$  is the rate of the code. Corruptions in codewords can either be in the form of erasures (missing entries, denoted by the symbol  $\perp$ ) or in the form of errors (wrong values from  $\mathbb{F}_2$ ).

Recall that a local list decoder outputs a list of algorithms which give oracle access to decoded messages or, in other words *implicitly compute* the decoded messages. This, and the notion of local list erasure-decoders are formalized in the following definitions.

► **Definition 1.1** (Implicit Computation). An algorithm  $A$  is said to implicitly compute  $x \in \mathbb{F}_2^n$  if, for all  $i \in [n]$ , the algorithm  $A$  on input  $i$ , outputs the  $i^{\text{th}}$  bit of  $x$ .

► **Definition 1.2** (Locally List Erasure-Decodable Codes (LLEDs)). A family of codes  $\{C_n : \mathbb{F}_2^n \rightarrow \mathbb{F}_2^N\}_{n \in \mathbb{N}}$  is  $(\alpha, q, L)$ -*locally list erasure-decodable* if there exists a randomized algorithm  $A$  such that, for every  $n \in \mathbb{N}$  and every  $w \in (\mathbb{F}_2 \cup \{\perp\})^N$  with at most an  $\alpha$  fraction of erasures, the algorithm  $A$  makes at most  $q$  queries to  $w$  and outputs a list of randomized algorithms  $\{T_1, T_2, \dots, T_L\}$  such that the following hold:

1. With probability at least  $2/3$ , for all  $x \in \mathbb{F}_2^n$  such that  $C_n(x)$  agrees with  $w$  on all nonerased bits, there exists an index  $j \in [L]$  such that  $T_j$  with oracle access to  $w$  implicitly computes  $x$ .
2. For all  $j \in [L]$  and  $i \in [n]$ , the expected number of queries that the algorithm  $T_j$  makes to  $w$  on input  $i$  is at most  $q$ .

The definition of an  $(\alpha, q, L)$ -LLDC is identical to Definition 1.2 except that the input word has no erasures, and the list is required to contain, with probability at least  $2/3$ , algorithms that implicitly compute messages corresponding to codewords disagreeing with the input word on at most an  $\alpha$  fraction of bits. The celebrated Goldreich-Levin theorem [19] states that the Hadamard code, defined next, is an LLDC that has an efficient decoder.

► **Definition 1.3** (Hadamard code). For  $a \in \mathbb{F}_2^n$ , let  $H_a : \mathbb{F}_2^n \rightarrow \mathbb{F}_2$  be defined as follows:  $H_a(x) = \bigoplus_{i \in [n]} a_i \cdot x_i$  for all  $x \in \mathbb{F}_2^n$ . The Hadamard code, denoted by  $\{\mathcal{H}_n : \mathbb{F}_2^n \rightarrow \mathbb{F}_2^{2^n}\}_{n \in \mathbb{N}}$ , is such that for  $a \in \mathbb{F}_2^n$ , the encoding  $\mathcal{H}_n(a)$  is the string of evaluations of  $H_a$  over  $\mathbb{F}_2^n$ .

Our first result is about the local list erasure-decodability of the Hadamard code. It is an analogue of the Golreich-Levin Theorem [19] for corruptions in the form of erasures.

► **Theorem 1.4** (Local List Erasure-Decoder for Hadamard). *There is an  $(\alpha, \Theta(\frac{1}{1-\alpha}), \Theta(\frac{1}{1-\alpha}))$ -local list erasure-decoder for the Hadamard code that works for every  $\alpha \in [0, 1)$ .*

The Goldreich-Levin theorem holds for any fraction of errors in  $[0, 1/2)$ . In contrast, our local list erasure-decoder works for any fraction of erasures less than 1. However, it is impossible to decode the Hadamard code in the presence of  $1/2$  fraction of errors because every Hadamard codeword has relative distance at most  $1/2$  from the all-zero codeword. Another improvement in Theorem 1.4 as compared to Golreich-Levin is in the list size and the query complexity: from  $\Theta(\frac{1}{(1/2-\alpha)^2})$  to  $\Theta(\frac{1}{1-\alpha})$ . Such an improvement is impossible if we are decoding against errors as opposed to erasures. Specifically, for the list size, Blinovskiy [8] and Guruswami and Vadhan [26] show that every list decoder for every binary code that is list decodable in the presence of an  $\alpha$  fraction of errors must output lists of size  $\Omega(\frac{1}{(1/2-\alpha)^2})$ . Grinberg, Shaltiel, and Viola [21] show that the same lower bound holds for query complexity.

### Separation between Erasure-Resilient and Tolerant Testing

We first describe the erasure-resilient and tolerant models of testing. A *property*  $\mathcal{P}$  is a set of strings. Given  $\alpha \in [0, 1)$ , a string is  $\alpha$ -erased if at most an  $\alpha$  fraction of its values are erasures (denoted by  $\perp$ ). A *completion* of an  $\alpha$ -erased string  $x \in \{0, 1, \perp\}^n$  is a string  $y \in \{0, 1\}^n$  that agrees with  $x$  on all the positions where  $x$  is nonerased. An  $\alpha$ -erasure-resilient  $\varepsilon$ -tester [12] for a property  $\mathcal{P}$  is a randomized algorithm that, given parameters  $\alpha \in [0, 1), \varepsilon \in (0, 1)$  and oracle access to an  $\alpha$ -erased string  $x$ , accepts with probability at least  $2/3$  if  $x$  has a completion in  $\mathcal{P}$  and rejects with probability at least  $2/3$  if, in every completion of  $x$ , at least an  $\varepsilon$  fraction of the nonerased values has to be changed to get a string in  $\mathcal{P}$ . The property  $\mathcal{P}$  is  $\alpha$ -erasure-resiliently  $\varepsilon$ -testable if there exists an  $\alpha$ -erasure-resilient  $\varepsilon$ -tester for  $\mathcal{P}$  with query complexity that depends only on the parameters  $\alpha$  and  $\varepsilon$  (but not on  $n$ ).

A string  $x \in \{0, 1\}^n$  is  $\varepsilon'$ -far ( $\alpha$ -close) from (to, respectively) a property  $\mathcal{P}$ , if the normalized Hamming distance of  $x$  from  $\mathcal{P}$  is at least  $\varepsilon'$  (at most  $\alpha$ , respectively). An  $(\alpha, \varepsilon')$ -tolerant tester [34] for  $\mathcal{P}$  is a randomized algorithm that, given parameters  $\alpha \in (0, 1), \varepsilon' \in (\alpha, 1)$  and oracle access to a string  $x$ , accepts with probability at least  $\frac{2}{3}$  if  $x$  is  $\alpha$ -close to  $\mathcal{P}$  and rejects with probability at least  $\frac{2}{3}$  if  $x$  is  $\varepsilon'$ -far from  $\mathcal{P}$ . The property  $\mathcal{P}$  is  $(\alpha, \varepsilon')$ -tolerantly testable if there exists an  $(\alpha, \varepsilon')$ -tolerant tester for  $\mathcal{P}$  with query complexity that depends only on  $\alpha$  and  $\varepsilon'$  (but not  $n$ ).

**Comparison of parameters.** We remark that, while comparing the two models, it is appropriate to compare  $(\alpha, \alpha + \varepsilon(1 - \alpha))$ -tolerant testing of a property  $\mathcal{P}$  with  $\alpha$ -erasure-resilient  $\varepsilon$ -testing of  $\mathcal{P}$  for the same values of  $\alpha$  and  $\varepsilon$ . The parameter  $\alpha$  in both models is an upper bound on the fraction of corruptions (erasures, or errors) that an adversary can make to an input. An  $\alpha$ -erasure-resilient  $\varepsilon$ -tester rejects with probability at least  $\frac{2}{3}$  if, for every way of completing an input string, one needs to change at least an  $\varepsilon$  fraction of the remaining part of the input to make it satisfy  $\mathcal{P}$ . Similarly, an  $(\alpha, \alpha + \varepsilon(1 - \alpha))$ -tolerant tester rejects with probability at least  $\frac{2}{3}$  if, for every way of *correcting* an  $\alpha$  fraction of the input values, one needs to change at least an  $\varepsilon$  fraction of the remaining  $(1 - \alpha)$  fraction of the input to make it satisfy  $\mathcal{P}$ .

**Separation.** The following theorem states that there exists a property that is erasure-resiliently testable but is not tolerantly testable. This proves that tolerant testing is, in general, harder problem than erasure-resilient testing.

- **Theorem 1.5 (Separation).** *There exist a property  $\mathcal{P}$  and constants  $\varepsilon, \alpha \in (0, 1)$  such that*
- $\mathcal{P}$  is  $\alpha$ -erasure-resiliently  $\varepsilon$ -testable;
  - $\mathcal{P}$  is not  $(\alpha, \alpha + \varepsilon(1 - \alpha))$ -tolerantly testable.

**Approximate Local List Erasure-Decoding and Strengthened Separation.** We obtain a separation better than in Theorem 1.5 with the help of a variant of LLEDCs, called approximate locally list erasure-decodable codes (ALLEDC). An approximate local list erasure-decoder is identical to a local list erasure-decoder in all aspects except that the algorithms in its list are required to implicitly compute strings that are just “close” to the actual messages. More formally,  $(\alpha, \beta, q, L)$ -ALLEDCs are defined as  $(\alpha, q, L)$ -LLEDCs in Definition 1.2, except that we replace “implicitly computes  $x$ ” at the end of Item 1 with “implicitly computes a string  $x' \in \mathbb{F}_2^n$  that is  $\beta$ -close to  $x$ ”.

The definition of an  $(\alpha, \beta, q, L)$ -approximate locally list decodable code (ALLDC) is identical to that of an  $(\alpha, \beta, q, L)$ -ALLEDC except that the input word has no erasures, and the list is required to contain, with probability at least  $2/3$ , algorithms that implicitly compute strings that are  $\beta$ -close to messages corresponding to codewords which are  $\alpha$ -close to the input word. We observe (Observation 4.2) that every  $(\alpha, \beta, q, L)$ -ALLDC is also a  $(2\alpha, \beta, 4q, 4L)$ -ALLEDC, and combine this observation with existing constructions for ALLDCs [28, 4] to obtain efficient ALLEDCs. We use them and get our strengthened separation.

- **Theorem 1.6 (Strengthened Separation).** *There exist a property  $\mathcal{P}'$  and constants  $\varepsilon, \alpha \in (0, 1)$  such that*
- $\mathcal{P}'$  is  $\alpha$ -erasure-resiliently  $\varepsilon$ -testable;
  - every  $(\alpha, \alpha + \varepsilon(1 - \alpha))$ -tolerant tester for  $\mathcal{P}'$  makes  $n^{\Omega(1)}$  queries.

### Relationship between Local Erasure-Decoding and Local Decoding.

We investigate the general relationship between the erasures and errors in the context of local unique and list decoding. We show that local (unique) decoding from erasures implies local (unique) decoding from errors, up to some loss in parameters.

► **Definition 1.7 (Locally Erasure-Decodable Codes (LEDCs)).** A code family  $\{C_n : \mathbb{F}_2^n \rightarrow \mathbb{F}_2^N\}_{n \in \mathbb{N}}$  is  $(\alpha, q)$ -locally erasure-decodable if there exists an algorithm  $A$  that, given an index  $i \in [n]$  and oracle access to an input word  $w \in (\{\perp\} \cup \mathbb{F}_2)^N$  with at most  $\alpha$  fraction of erasures, makes at most  $q$  queries to  $w$  and outputs  $x_i$  with probability at least  $\frac{2}{3}$ .

An  $(\alpha, q)$ -locally decodable code (LDC) is defined similarly to an  $(\alpha, q)$ -LEDC except that the input word  $w$  contains at most  $\alpha$  fraction of errors instead of erasures. We observe (Observation 6.4) that an LDC is also locally erasure-decodable from (nearly) twice as many erasures. We also show that constant-query LEDCs are constant-query locally decodable (up to constant loss in parameters).

► **Theorem 1.8.** *For every  $\alpha \in [0, 1)$ , if a code family  $\{C_n : \mathbb{F}_2^n \rightarrow \mathbb{F}_2^N\}_{n \in \mathbb{N}}$  is  $(\alpha, q)$ -locally erasure-decodable, then it is  $(\frac{\alpha}{O(q^2 \cdot 81^q)}, O(q \cdot 9^q))$  locally decodable.*

We note that although our final code has small decoding radius (that is, it tolerates only a small fraction of errors), the decoding radius can be amplified to any constant arbitrarily close to  $1/4$  at the cost of increasing the query complexity and encoding length by a constant factor. Specifically, using a local version of the AEL transformation [1] (see [30, Lemma 3.1]), one can amplify the decoding radius to any constant arbitrarily close to  $1/2$  at the cost of increasing the query complexity, alphabet size, and length by constant factors. The alphabet then can be reduced back to binary by encoding the binary representation of each alphabet symbol with the Hadamard code. The length will grow by another constant factor, and using a local version of the GMD decoder [30, Corollary 3.9], one can show that final decoding radius is arbitrarily close to  $1/4$  and query complexity grows only by a constant factor.

## 1.2 Open Questions

The main open question raised by our work is whether local list decoding is significantly easier in terms of the query complexity, the list size, or the rate of codes when corruptions are in the form of erasures. The same question can be asked about approximate local list decoding. Our local list erasure-decoder for the Hadamard code shows that there is some advantage for having erasures over errors, in terms of the list size and query complexity, for some settings of parameters. A positive or negative answer to this question, combined with our result on the equivalence of errors and erasures in the local decoding regime, will enhance the understanding of whether local list decoding is an inherently more powerful model when compared to local decoding.

## 2 Local List Erasure-Decoding of the Hadamard Code

In this section, we describe a local list erasure-decoder for the Hadamard code and prove Theorem 1.4. We follow the style of the proof of the Goldreich-Levin theorem given in a tutorial by Luca Trevisan [39] on the applications of coding theory to complexity.

**Proof of Theorem 1.4.** For  $b_1, b_2 \in \mathbb{F}_2$ , let  $b_1 \oplus b_2$  denote the XOR of  $b_1$  and  $b_2$ . For vectors  $x, y \in \mathbb{F}_2^n$ , let  $x \odot y$  denote the bitwise XOR of  $x$  and  $y$ . Let  $e_k \in \mathbb{F}_2^n$  denote the  $k^{\text{th}}$  standard basis vector. A codeword of the Hadamard code  $\mathcal{H}_n$  (see Definition 1.3) is the string of all evaluations of a linear function mapping  $\mathbb{F}_2^n$  to  $\mathbb{F}_2$ . A function  $f : \mathbb{F}_2^n \rightarrow \mathbb{F}_2 \cup \{\perp\}$  is  $\alpha$ -erased, if  $f$  evaluates to  $\perp$  on at most  $\alpha$  fraction of its domain. Our local list erasure-decoder, described in Algorithm 1, gets a parameter  $\alpha \in [0, 1)$  as its input and has oracle access to an  $\alpha$ -erased linear function  $f : \mathbb{F}_2^n \rightarrow \mathbb{F}_2 \cup \{\perp\}$  (or, equivalently, oracle access to an  $\alpha$ -erased codeword of  $\mathcal{H}_n$ ).

Let  $t = \lceil \log_2(1 + \frac{12}{1-\alpha}) \rceil$ . Consider  $z_1, z_2, \dots, z_t \in \mathbb{F}_2^n$  sampled uniformly and independently at random. For a nonempty set  $S \subseteq [t]$ , let  $z_S$  denote  $\bigodot_{i \in S} z_i$ . Let  $z_\emptyset$  denote  $\vec{0}$ . For any two nonempty sets  $R, S \subseteq [t]$  such that  $R \neq S$ , the vectors  $z_R$  and  $z_S$  are independently and uniformly distributed in  $\mathbb{F}_2^n$ . Recall that for a string  $a \in \mathbb{F}_2^n$ , let  $H_a : \mathbb{F}_2^n \rightarrow \{0, 1\}$  denote the Hadamard encoding (see Definition 1.3) of  $a$ .

Let  $a \in \mathbb{F}_2^n$  be such that for all  $x \in \mathbb{F}_2^n$  where  $f(x) \neq \perp$ , the functions  $H_a$  and  $f$  agree with each other. There exists some iteration of Step 5 of Algorithm 1 such that  $b_i = H_a(z_i)$  for all  $i \in B$ . Let  $T$  and  $A$  denote the algorithms whose descriptions are generated in Steps 12 and 7 of this iteration respectively.

First, we show that for  $x$  distributed uniformly in  $\mathbb{F}_2^n$ , the algorithm  $A$  on input  $x$ , returns  $H_a(x)$  with probability at least  $\frac{2}{3}$ . Fix  $x \in \mathbb{F}_2^n$ . Consider a set  $S \subseteq [t]$  such that  $f(x \odot z_S) \neq \perp$ . According to the description of  $A$ , we get,  $A(x) = (\bigoplus_{j \in S \cap B} b_j) \oplus$

**Algorithm 1** Local List Erasure-Decoder for the Hadamard code.

---

**Input:**  $\alpha \in [0, 1)$ ; oracle access to  $\alpha$ -erased linear function  $f : \mathbb{F}_2^n \rightarrow \mathbb{F}_2 \cup \{\perp\}$

- 1: Let  $t = \lceil \log_2(1 + \frac{12}{1-\alpha}) \rceil$ .
- 2: Choose  $z_1, z_2, \dots, z_t \in \mathbb{F}_2^n$  uniformly and independently at random.
- 3: Let  $z_S \leftarrow \bigodot_{i \in S} z_i$  for all nonempty  $S \subseteq [t]$ . Let  $z_\emptyset \leftarrow \vec{0}$ .
- 4: Set  $B \leftarrow \{i \in [t] : f(z_i) = \perp\}$ .
- 5: **for** all  $b_1, b_2, \dots, b_{|B|} \in \{0, 1\}$  **do** **define**
- 6: ▷ Description of the local decoder  $T_{b_1, \dots, b_{|B|}}$  follows.
- 7:     **function**  $A_{b_1, \dots, b_{|B|}}$
- 8:         **input:**  $x \in \mathbb{F}_2^n$ ; oracle access to  $f : \mathbb{F}_2^n \rightarrow \mathbb{F}_2 \cup \{\perp\}$
- 9:         **for** all  $S \subseteq [t]$  **do**
- 10:             **if**  $f(x \odot z_S) \neq \perp$  **then return**  $(\bigoplus_{j \in S \cap B} b_j) \oplus (\bigoplus_{j \in S \cap ([t] \setminus B)} f(z_j)) \oplus f(x \odot z_S)$ .
- 11:         **Return**  $\perp$ .
- 12:     **function**  $T_{b_1, \dots, b_{|B|}}$
- 13:         **input:**  $k \in [n]$ ; oracle access to  $f : \mathbb{F}_2^n \rightarrow \mathbb{F}_2 \cup \{\perp\}$
- 14:         **repeat**
- 15:             Pick  $y \in \mathbb{F}_2^n$  uniformly and independently at random.
- 16:              $u \leftarrow A_{b_1, \dots, b_{|B|}}(y \odot e_k)$ ,  $v \leftarrow A_{b_1, \dots, b_{|B|}}(y)$ .
- 17:             **if**  $v \neq \perp$  and  $u \neq \perp$  **then return**  $u \oplus v$ .
- 18: **Return** the descriptions of  $T_{b_1, \dots, b_{|B|}}$  for all  $b_1, b_2, \dots, b_{|B|} \in \{0, 1\}$ .

---

$$\left(\bigoplus_{j \in S \cap ([t] \setminus B)} f(z_j)\right) \oplus f(x \odot z_S) = \left(\bigoplus_{j \in S \cap B} H_a(z_j)\right) \oplus \left(\bigoplus_{j \in S \cap ([t] \setminus B)} H_a(z_j)\right) \oplus H_a(x \odot z_S) = \left(\bigoplus_{j \in S} H_a(z_j)\right) \oplus H_a(x) = H_a(x).$$

Let  $\alpha^* \leq \alpha$  denote the fraction of erasures in  $f$ . For each  $S \subseteq [t]$  and  $x \in \mathbb{F}_2^n$ , we have that  $f(x \odot z_S) \neq \perp$  with probability equal to  $1 - \alpha^*$ , since  $x \odot z_S$  is uniformly distributed in  $\mathbb{F}_2^n$ . Define an indicator random variable  $Z_S = \mathbb{1}(f(x \odot z_S) \neq \perp)$ . Then  $\mathbb{E}[Z_S] = 1 - \alpha^*$  and  $\text{Var}(Z_S) = (1 - \alpha^*) \cdot \alpha^*$ . Note that the collection  $\{x \odot z_S | S \subseteq [t], S \neq \emptyset\}$  is pairwise independent, and hence the collection  $\{Z_S | S \subseteq [t], S \neq \emptyset\}$  is also pairwise independent.

Let  $Z = \sum_{S \subseteq [t], S \neq \emptyset} Z_S$ . The random variable  $Z$  denotes the number of nonerased values among  $f(x \odot z_S)$  over all nonempty  $S \subseteq [t]$ . The event that  $\forall S \subseteq [t], S \neq \emptyset, f(x \odot z_S) = \perp$  is equivalent to the event that  $Z < 1$ . Also,  $\mathbb{E}[Z] = \sum_{S \subseteq [t], S \neq \emptyset} \mathbb{E}[Z_S] = (1 - \alpha^*) \cdot (2^t - 1)$  and  $\text{Var}[Z] = \sum_{S \subseteq [t], S \neq \emptyset} \text{Var}[Z_S] = (2^t - 1) \cdot \alpha^*(1 - \alpha^*)$ . By Chebyshev's inequality,

$$\begin{aligned} \Pr[Z < 1] &= \Pr[\mathbb{E}[Z] - Z > \mathbb{E}[Z] - 1] \\ &\leq \Pr\left[\mathbb{E}[Z] - Z > \frac{(1 - \alpha^*) \cdot (2^t - 1)}{2}\right] \leq \frac{4\text{Var}(Z)}{(1 - \alpha^*)^2 \cdot (2^t - 1)^2} \leq \frac{1}{3}. \end{aligned}$$

The last inequality follows from our setting of  $t$ . Therefore, for  $x$  distributed uniformly in  $\mathbb{F}_2^n$ , the algorithm  $A$  on input  $x$ , returns  $H_a(x)$  with probability at least  $\frac{2}{3}$ .

We now prove that  $T$  implicitly computes  $a \in \mathbb{F}_2^n$  and that the expected number of queries that it makes to  $f$  is  $\Theta(\frac{1}{1-\alpha})$ . It is clear that the output of  $T$  on input  $k$  is always  $a[k] = H_a(y \odot e_k) \oplus H_a(y) = H_a(e_k)$ . The number of queries made by  $T$  to  $A$  is a geometric random variable with success probability  $\geq \frac{1}{3}$ . Hence, the expected number of queries made by  $T$  to  $A$  is at most 3. Since the query complexity of  $A$  is at most  $2^t$ , the expected number of queries made to  $f$  in one invocation of  $T$  is  $\Theta(2^t)$ , that is,  $\Theta(\frac{1}{1-\alpha})$ . The number of algorithms whose descriptions are generated is also at most  $2^t$ , which is,  $\Theta(\frac{1}{1-\alpha})$ . ◀



### 3 Separation

In this section, we describe a property  $\mathcal{P}$  that is erasure-resiliently testable using a constant number of queries, but not tolerantly testable using a constant number of queries, and prove Theorem 1.5. In fact, we prove the following (more general) statement and show that it implies Theorem 1.5.

► **Theorem 3.1.** *Let  $\varepsilon^* \in (0, \frac{1}{100})$  be a constant. There exists a property  $\mathcal{P} \subseteq \{0, 1\}^*$  such that*

- *for every  $\alpha \in [0, \frac{3\varepsilon^*}{16})$  and  $\varepsilon \in (\frac{3\varepsilon^*}{4(1-\alpha)}, 1)$ , the property  $\mathcal{P}$  can be  $\alpha$ -erasure-resiliently  $\varepsilon$ -tested using  $O(\frac{1}{\varepsilon(1-2\alpha)})$  queries.*
- *for all  $\alpha \in (\frac{\varepsilon^*}{8}, 1)$  and  $\varepsilon' \in (\alpha, \varepsilon^* - \frac{(\varepsilon^*)^2}{4})$ , the query complexity of  $(\alpha, \varepsilon')$ -tolerant testing  $\mathcal{P}$  on inputs of length  $N$  is  $\tilde{\Omega}(\log N)$ .*

#### 3.1 Description of the Separating Property $\mathcal{P}$

The property  $\mathcal{P}$  is defined in terms of a property  $\mathcal{R}$  that is hard to test in the standard property testing model [18, 37], a probabilistically checkable proof system (PCP of proximity [6, 11]<sup>2</sup>) for the problem of testing  $\mathcal{R}$ , and the Hadamard code. We discuss them below. The idea of using PCPs of proximity in separating property testing models comes from the work of Fischer and Fortnow [15]. Our contribution is to use locally list decodable codes in this context.

Given a Boolean formula  $\phi$  over  $n$  variables, let  $\mathcal{R}_\phi \subseteq \{0, 1\}^n$  denote the set of all satisfying assignments to  $\phi$ , represented as  $n$ -bit strings. Ben-Sasson, Harsha and Raskhodnikova [7] showed that for infinitely many  $n \in \mathbb{N}$ , there exists a 3CNF formula  $\phi_n$  on  $n$  variables such that every tester for  $\mathcal{R}_{\phi_n}$  requires  $\Omega(n)$  queries.

► **Lemma 3.2** ([7]). *There exists a parameter  $\varepsilon^* \in (0, 1)$  and a countably infinite set  $\aleph \subseteq \mathbb{N}$  such that for all  $n \in \aleph$ , there exists a 3CNF formula  $\phi_n$  with  $n$  variables and  $\Theta(n)$  clauses such that every  $\varepsilon^*$ -tester for  $\mathcal{R}_{\phi_n}$  has query complexity  $\Omega(n)$ .*

As mentioned before, another important ingredient in the description of the separating property  $\mathcal{P}$  is a probabilistically checkable proof system for property testing problems, called PCP of proximity, defined and studied independently by Ben-Sasson et al. [6] and Dinur and Reingold [11]. PCPs of proximity were further studied by Dinur [10] and Meir [32, 33].

► **Definition 3.3** (PCP of proximity [6, 11]). Given a property  $P_n \subseteq \{0, 1\}^n$ , the PCP of proximity (PCPP) for  $P_n$  is a randomized algorithm  $V$  that takes a parameter  $\varepsilon \in (0, 1]$  as input, gets oracle access to a string  $y \circ \pi$ , where  $y \in \{0, 1\}^n$  is the input and  $\pi \in \{0, 1\}^m$  is the proof, and satisfies the following:

- if  $y \in P_n$ , then, for some  $\pi$ , the algorithm  $V$  always accepts  $y \circ \pi$ ;
- if  $y$  is  $\varepsilon$ -far from  $P_n$ , then, for every  $\pi$ , the algorithm  $V$  rejects  $y \circ \pi$  with probability at least  $\frac{2}{3}$ .

A result by Dinur [10, Corollary 8.4] states that there are efficient PCPPs (over a small constant alphabet  $\Sigma$ ) for testing properties (over  $\Sigma$ ) that are decidable using polynomial-sized circuits. By representing the symbols in  $\Sigma$  using the binary alphabet, we obtain the following.

<sup>2</sup> PCPs of proximity are referred to as assignment testers by Dinur and Reingold [11]. Ben-Sasson et al. [6] and Dinur and Reingold [11] defined these objects concurrently and independently in order to obtain simpler and more efficient PCP constructions.



► **Lemma 3.4** ([10]). *If  $P_n \subseteq \{0,1\}^n$  is a property decidable by a circuit of size  $s(n)$ , then there exists a PCPP  $V$  that works for every  $\varepsilon \in (0,1]$ , uses a proof of length at most  $s(n) \cdot \text{polylog } s(n)$ , and has query complexity  $O(\frac{1}{\varepsilon})$ . Moreover, the queries of  $V$  are nonadaptive.*

► **Claim 3.5.** *There exists a constant  $c > 0$  such that for every large enough  $n \in \mathbb{N}$ , there exists a PCPP  $V$  for the property  $\mathcal{R}_{\phi_n}$  that works for all  $\varepsilon \in (0,1]$ , uses a proof of length at most  $cn \cdot \text{polylog } n$ , and has query complexity  $O(\frac{1}{\varepsilon})$ .*

The following is the definition of our separating property  $\mathcal{P}$ . At a high level, the definition says that, for all  $n \in \mathbb{N}$ , a string of length  $O(2^{n \cdot \text{polylog } n})$  satisfies  $\mathcal{P}$  if its first part is the repetition of a string  $y$  satisfying  $\mathcal{R}$ , and the second part is the encoding (by the Hadamard code) of  $y$  concatenated with a proof  $\pi$  that makes the algorithm  $V$  in Claim 3.5 accept.

► **Definition 3.6 (Separating Property  $\mathcal{P}$ ).** Let  $\varepsilon^* \in (0,1)$  be as in Lemma 3.2. For  $n \in \mathbb{N}$ , let  $p(n) \leq cn \cdot \text{polylog } n$  denote the length of proof that the algorithm  $V$  in Claim 3.5 has oracle access to. A string  $x \in \{0,1\}^N$  of length  $N = \frac{4}{\varepsilon^*} \cdot 2^{n+p(n)}$  satisfies  $\mathcal{P}$  if:

1. The first  $(\frac{4}{\varepsilon^*} - 1) \cdot 2^{n+p(n)}$  bits of  $x$  (called the *plain part* of  $x$ ) consist of  $(\frac{4}{\varepsilon^*} - 1) \cdot \frac{2^{n+p(n)}}{n}$  repetitions of a string  $y \in \mathcal{R}_{\phi_n}$  of length  $n$ , for  $\phi_n$  from Lemma 3.2.
2. The remaining bits of  $x$  (called the *encoded part* of  $x$ ) form the Hadamard encoding of a string  $y \circ \pi(y)$  of length  $n + p(n)$ , where  $\circ$  denotes the concatenation operation on strings. The string  $y \in \{0,1\}^n$  is the same as the one in the description of the plain part. The string  $\pi(y) \in \{0,1\}^{p(n)}$  is a proof such that the algorithm  $V$  (from Claim 3.5) accepts when given oracle access to  $y$  and  $\pi(y)$ .

## 3.2 Proof of Theorem 3.1

In this section, we prove Theorem 3.1, which in turn implies Theorem 1.5. Lemmas 3.7 and 3.10 prove the first and second parts of Theorem 3.1, respectively. The erasure-resilient tester for  $\mathcal{P}$  first obtains a list of (implicit) decodings of the encoded part (see Definition 3.6) of an input string  $x \in \{0,1\}^N$  using the local list erasure-decoder guaranteed by Theorem 1.4. If  $x \in \mathcal{P}$ , with high probability, at least one of the algorithms implicitly computes (see Definition 1.1) the string  $y \circ \pi(y)$ , where  $y$  is such that the plain part of  $x$  (see Definition 3.6) consists of repetitions of  $y$ , and  $\pi(y)$  is a proof string such that the algorithm  $V$  (from Claim 3.5) accepts upon oracle access to  $y \circ \pi(y)$ . In case  $x$  is  $\varepsilon$ -far from  $\mathcal{P}$  we show that for every algorithm  $T$  output by the local list erasure-decoder, the string  $y' \circ \pi(y')$  implicitly computed by  $T$  is such that, (1) either the plain part of  $x$  is far from being the repetitions of  $y'$ , (2) or  $y'$  is far from  $\mathcal{R}$  (in which case, the algorithm  $V$  from Claim 3.5 rejects when given oracle access to  $y' \circ \pi(y')$ ). To show that tolerant testing of  $\mathcal{P}$  is hard, we reduce  $\varepsilon^*$ -testing of  $\mathcal{R}_{\phi_n}$  to it. Specifically, given oracle access to a string  $y \in \{0,1\}^n$  that we want to  $\varepsilon^*$ -test, we simulate oracle access to a string  $x \in \{0,1\}^N$  such that the plain part of  $x$  consists of repetitions of  $y$ , and every bit in the encoded part of  $x$  is 0. Since every Hadamard codeword has an equal number of 0s and 1s, the string  $x$  can be thought of as having 0.5 fraction of “errors” in the encoded part. If  $y \in \mathcal{R}_{\phi_n}$ , then the string  $x$  is close to being in  $\mathcal{P}$ , as the errors are only in the encoded part of  $x$  and the length of the encoded part is a small fraction of the length of  $x$ . If  $y$  is far from  $\mathcal{R}_{\phi_n}$ , then  $x$  is also far from  $\mathcal{P}$ , since the plain part of  $x$ , whose length is a large fraction of the length of  $x$ , is the repetitions of  $y$ . Thus, the decision of a tolerant tester for  $\mathcal{P}$  on  $x$  can be used to test  $y$  for  $\mathcal{R}_{\phi_n}$ , implying that the complexity of tolerant testing of  $\mathcal{P}$  is equal to the complexity of testing  $\mathcal{R}_{\phi_n}$ .

---

**Algorithm 2** Erasure-resilient tester for separating property  $\mathcal{P}$ .
 

---

**Input:**  $\alpha, \varepsilon \in (0, 1), N = \frac{4}{\varepsilon^*} \cdot 2^{(n+p(n))}$ ; oracle access to  $x \in \{0, 1, \perp\}^N$

- 1: Set  $s \leftarrow (\frac{4}{\varepsilon^*} - 1) \cdot 2^{(n+p(n))}/n$ ,  $\varepsilon' \leftarrow \varepsilon(1 - 2\alpha)/3$ .
- 2: Set  $Q \leftarrow C/(\varepsilon(1 - 2\alpha))$  for a large enough constant  $C$ .
- 3: **Accept** whenever the number of queries exceeds  $Q$ .
- 4: Let  $T_1, T_2, \dots, T_L$  be the list of algorithms returned by a  $(\frac{3}{4}, q, L)$ -local list erasure-decoder for the Hadamard code (Algorithm 1), given oracle access to  $x[sn + 1..N]$ , the encoded part of  $x$ .
- 5: **for** each  $k \in [L]$  **do**
- 6:    $\triangleright$  Check if the plain part of  $x$  is the repetition of  $y$ , where  $y$  denotes the first  $n$  bits of the decoding (given by  $T_k$ ) of the encoded part of  $x$ .
- 7:    **repeat**  $\lceil \frac{9 \log L}{\varepsilon(1-2\alpha)} \rceil$  times:
  - 8:      Pick  $a \in_R [n], i \in_R [s]$ .
  - 9:      **if**  $x[(i-1)n + a] \neq \perp$  and  $T_k(a) \neq x[(i-1)n + a]$  **then**
  - 10:        **Discard** the current  $k$
- 11:    $\triangleright$  Check if the string  $y \in \mathcal{R}_{\phi_n}$ , where  $y$  denotes the first  $n$  bits of the decoding (by  $T_k$ ) of the encoded part of  $x$ .
- 12:    **repeat**  $\lceil 4 \log L \rceil$  times:
  - 13:      Run  $V$ , from Claim 3.5, with input  $\varepsilon'$  and oracle access to  $T_k$ .
  - 14:      **Discard** the current  $k$  if  $V$  rejects.
- 15: **Reject** if every  $k \in [L]$  is **discarded**; otherwise, **accept**.

---

We first prove the existence of an efficient erasure-resilient tester for  $\mathcal{P}$ . An  $\alpha$ -erased string  $x$  is  $\varepsilon$ -far from a property  $\mathcal{P}$  if there is no way to complete  $x$  to a string that satisfies  $\mathcal{P}$  without changing at least an  $\varepsilon$  fraction of the nonerased values in  $x$ .

► **Lemma 3.7.** *Let  $\varepsilon^* \in (0, 1)$  be as in Lemma 3.2. For every  $\alpha \in [0, \frac{3\varepsilon^*}{16})$ , and every  $\varepsilon \in (\frac{3\varepsilon^*}{4(1-\alpha)}, 1)$ , the property  $\mathcal{P}$  can be  $\alpha$ -erasure-resiliently  $\varepsilon$ -tested using  $O(\frac{1}{\varepsilon(1-2\alpha)})$  queries.*

**Proof.** The erasure-resilient tester for  $\mathcal{P}$  is described in Algorithm 2. The query complexity of the tester is evident from its description. We now prove that the tester, with probability at least  $\frac{2}{3}$ , accepts strings in  $\mathcal{P}$  and rejects strings that are  $\varepsilon$ -far from  $\mathcal{P}$ .

Let  $\aleph, \varepsilon^* \in (0, 1)$  be as in Lemma 3.2. Fix  $n \in \aleph$  and let  $p(n)$  and  $N$  be as in Definition 3.6. Let  $s$  denote  $(\frac{4}{\varepsilon^*} - 1) \cdot \frac{2^{n+p(n)}}{n}$ . Consider a string  $x \in \{0, 1\}^N$  that we want to erasure-resiliently test for  $\mathcal{P}$ . As in Definition 3.6, we refer to the substring  $x[1 \dots sn]$  as the plain part of  $x$  and the substring  $x[sn + 1 \dots N]$  as the encoded part of  $x$ .

Assume that  $x \in \mathcal{P}$ . Since  $\alpha < 3\varepsilon^*/16$ , the fraction of erasures in the encoded part of  $x$  is at most  $3/4$ . Hence, by Theorem 1.4, with probability at least  $2/3$ , there exists an algorithm  $T_k$  computed in Step 4 of Algorithm 2, such that  $T_k$  implicitly computes the string  $y \circ \pi \in \{0, 1\}^{n+p(n)}$ , where  $y \in \mathcal{R}_{\phi_n}$ , the plain part of  $x$  can be completed to a repetition of  $y$ , and  $\pi$  is a proof such that the algorithm  $V$  (from Claim 3.5) accepts when given oracle access to  $y \circ \pi$ . Therefore  $k$  is not discarded in either Step 10 or Step 14. Thus, the tester will accept with probability at least  $2/3$ .

Now, assume that  $x$  is  $\varepsilon$ -far from  $\mathcal{P}$ . Let  $E$  denote the event that the number of queries made by the tester does not exceed its query budget. We will first show that, conditioned on  $E$ , the tester rejects  $x$  with probability at least  $4/5$ .

Let  $\mathcal{N}_1$  denote the set of nonerased points in the plain part of  $x$  and  $\mathcal{N}_2$  denote those in the encoded part and let  $\mathcal{N}$  denote the set of nonerased points in  $x$ . Even if all of at most  $N\alpha$  erased points in  $x$  are in the plain part of  $x$ , the total number of nonerased points

in plain part of  $x$ , which is  $|\mathcal{N}_1|$ , is at least  $sn - N\alpha$ . Since, for large enough  $n$ , we have  $N \leq 2sn$  by Definition 3.6, we can see that  $|\mathcal{N}_1| \geq sn(1 - 2\alpha)$ . We first prove two claims about the plain part of  $x$ , that is,  $x[1 \dots sn]$ .

► **Claim 3.8.** *The plain part of  $x$  is  $\frac{2\varepsilon}{3}$ -far from being  $s$  repetitions of a string  $y \in \mathcal{R}_{\phi_n}$ .*

From Claim 3.8, it follows that at least  $\frac{2\varepsilon \cdot |\mathcal{N}_1|}{3} \geq \frac{2\varepsilon \cdot sn(1-2\alpha)}{3}$  nonerased points need to be changed in the plain part of  $x$  for it to be  $s$  repetitions of a string  $y \in \mathcal{R}_{\phi_n}$ .

► **Claim 3.9.** *For every  $y \in \{0, 1\}^n$ , if the plain part of  $x$  can be changed to  $s$  repetitions of  $y$  by modifying less than  $\frac{\varepsilon \cdot sn(1-2\alpha)}{3}$  nonerased values, then  $y$  is  $\frac{\varepsilon(1-2\alpha)}{3}$ -far from  $\mathcal{R}_{\phi_n}$ .*

Fix  $k \in [L]$ . Let  $y' \in \{0, 1\}^n$  be the first  $n$  bits from the left in the decoding, using  $T_k$ , of the encoded part of  $x$ . We will show that the algorithm discards  $k$  with high probability. We split the analysis into two cases.

**Case I:** Suppose we need to change at least  $\frac{\varepsilon |\mathcal{N}_1|}{3} \geq \frac{\varepsilon \cdot sn(1-2\alpha)}{3}$  nonerased points in the plain part of  $x$  for it to become  $s$  repetitions of  $y'$ . We show that in this case, Steps 7-10 discard  $k$  with probability at least  $\frac{9}{10L}$ . A point  $(i-1)n + a$  for  $i \in [s]$  and  $a \in [n]$  is called a witness if  $x[(i-1)n + a] \neq \perp$  and  $x[(i-1)n + a] \neq y'[a]$ . Since we need to change at least  $\varepsilon \cdot sn(1-2\alpha)/3$  nonerased points in the plain part of  $x$  for it to become  $s$  repetitions of  $y'$ , there are at least  $\varepsilon \cdot sn(1-2\alpha)/3$  witnesses in the plain part of  $x$ . In each iteration of Steps 7-10, the point selected is a witness with probability at least  $\frac{\varepsilon \cdot sn(1-2\alpha)}{3sn} = \frac{\varepsilon(1-2\alpha)}{3}$ . Thus, in  $\lceil \frac{9 \log L}{\varepsilon(1-2\alpha)} \rceil$  iterations, Algorithm 2 finds a witness (and discards  $k$ ) with probability at least  $9/10L$ .

**Case II:** In this case, we assume that we can change less than  $\varepsilon \cdot sn(1-2\alpha)/3$  nonerased points in the plain part of  $x$  and make it  $s$  repetitions of  $y'$ . Then, by Claim 3.9,  $y'$  is  $\varepsilon \cdot (1-2\alpha)/3$ -far from  $\mathcal{R}_{\phi_n}$ . Let  $\varepsilon' = \frac{\varepsilon(1-2\alpha)}{3}$ . By Claim 3.5, for every proof  $\pi \in \{0, 1\}^{p(n)}$ , the algorithm  $V$  (from Claim 3.5), on input  $\varepsilon'$  and oracle access to  $y' \circ \pi$  (obtained via  $T_k$ ), rejects (causing  $k$  to be discarded) with probability at least  $2/3$ . Thus, the probability that tester fails to discard  $k$  in  $\lceil 4 \log L \rceil$  independent iterations of Steps 12-14 is at most  $1/16L$ .

Therefore, the probability that the tester fails to discard  $k$  is at most  $\frac{1}{10L} + \frac{1}{16L} < \frac{1}{5L}$ . By the union bound, the probability that Algorithm 2 fails to discard some  $k \in [L]$  is at most  $1/5$ . Thus, conditioned on the event  $E$  that the number of queries made by the tester does not exceed its query budget, with probability at least  $4/5$ , the tester rejects.

We bound the probability of  $E$  by first showing that the expected number of queries made by Algorithm 2 is  $O(\frac{1}{\varepsilon(1-2\alpha)})$  and then applying Markov's inequality. Hence, the probability that the tester accepts  $x$  that is  $\varepsilon$ -far from  $\mathcal{P}$  is at most  $1/3$ . ◀

► **Lemma 3.10.** *Let  $\varepsilon^* \in (0, 1)$  be as in Lemma 3.2. For every  $\alpha \in (\frac{\varepsilon^*}{8}, 1)$  and  $\varepsilon' \in (\alpha, \varepsilon^* - \frac{(\varepsilon^*)^2}{4})$ , the query complexity of  $(\alpha, \varepsilon')$ -tolerant testing  $\mathcal{P}$  on strings of length  $N$  is  $\tilde{\Omega}(\log N)$ .*

**Proof.** Let  $\aleph, \varepsilon^* \in (0, 1)$  be as in Lemma 3.2. We will prove the lemma by showing a reduction from  $\varepsilon^*$ -testing of  $\mathcal{R}_{\phi_n}$ . Fix  $n \in \aleph$  and let  $p(n)$  and  $N$  be as in Definition 3.6. Let  $s$  denote  $(\frac{4}{\varepsilon^*} - 1) \cdot \frac{2^{n+p(n)}}{n}$ .

Consider a string  $y \in \{0, 1\}^n$  that we want to  $\varepsilon^*$ -test for  $\mathcal{R}_{\phi_n}$ . Let  $x \in \{0, 1\}^N$  be the string where the first  $sn$  bits of  $x$  are  $s$  repetitions of  $y$  and the remaining bits are all 0s. We refer to the substring  $x[1 \dots sn]$  as the plain part of  $x$  and the substring  $x[sn + 1 \dots N]$  as the encoded part of  $x$ .

Assume that  $A$  is an  $(\alpha, \varepsilon')$ -tolerant tester for  $\mathcal{P}$ . We now describe an  $\varepsilon^*$ -tester  $A'$  for  $\mathcal{R}_{\phi_n}$  that has the same query complexity as  $A$ . Given oracle access to  $y \in \{0, 1\}^n$ , the tester  $A'$  runs the tester  $A$  on the string  $x \in \{0, 1\}^N$  and accepts if and only if  $A$  accepts, where  $x$  is constructed from  $y$  as described above. Observe that one can simulate a query to  $x$  by making at most one query to  $y$ .

If  $y \in \mathcal{R}_{\phi_n}$ , then  $x$  is  $\alpha$ -close to  $\mathcal{P}$ . Observe that the encoded part of  $x$  needs to be changed in at most  $1/2$  fraction of its positions in order to make it the encoding of a string  $y \circ \pi$ , where  $\pi$  is a proof that makes a PCP of proximity for testing  $\mathcal{R}_{\phi_n}$  accept. This follows from the fact that the normalized weight of every nonzero codeword in the Hadamard code is  $1/2$ . Thus, the fraction of bits in  $x$  that needs to be changed in order to make it satisfy  $\mathcal{P}$  is at most  $\frac{1}{2} \cdot \frac{N-sn}{N} = \frac{\varepsilon^*}{8}$ , which is less than  $\alpha$ . Therefore, by definition,  $A'$  will accept  $x$  with probability at least  $2/3$ .

If  $y$  is  $\varepsilon^*$ -far from  $\mathcal{R}_{\phi_n}$ , then  $x$  needs to be changed in at least  $\varepsilon^* \cdot sn$  positions to make it satisfy  $\mathcal{P}$ . From this, one can observe that  $x$  is  $(\varepsilon^* - \frac{(\varepsilon^*)^2}{4})$ -far from  $\mathcal{P}$ . Hence, for all  $\varepsilon' < \varepsilon^* - \frac{(\varepsilon^*)^2}{4}$ , we have that  $A$  will reject  $x$  with probability at least  $2/3$ , and therefore  $A'$  will reject  $y$  with probability at least  $2/3$ .

Thus, we have shown that the query complexity of  $(\alpha, \varepsilon')$ -tolerant testing  $\mathcal{P}$  is at least the query complexity of  $\varepsilon^*$ -testing  $\mathcal{R}_{\phi_n}$ . Hence, the query complexity of  $(\alpha, \varepsilon')$ -tolerant testing  $\mathcal{P}$  is  $\Omega(n)$ , which is equal to  $\tilde{\Omega}(\log N)$ . ◀

**Proof of Theorem 1.5.** For every  $0 < \varepsilon^* < 1/100$ , the system of constraints on  $\alpha, \varepsilon \in (0, 1)$  (by Theorem 3.1) has a feasible solution with  $\alpha = \varepsilon^*/6$  and  $\varepsilon = 4\varepsilon^*/5$ . ◀

## 4 Approximate Local List Erasure-Decoding

In this section, we prove the existence of an approximate locally list erasure-decodable code (ALLEDC) with inverse polynomial rate. Our starting point is an approximate locally list decodable code (ALLDC) due to Impagliazzo et al. [28]. To this code, we apply Observation 4.2 which states that every ALLDC that works in the presence of errors also works in the presence of twice as many erasures (with the same parameters up to constant factors).

► **Theorem 4.1** ([28] as restated by [4]). *For every  $\gamma, \beta > 0$ , there exists a number  $f(\gamma, \beta) > 0$  and a code family  $\{C_k : \mathbb{F}_2^k \rightarrow \mathbb{F}_2^{f(\gamma, \beta)k^5}\}_{k \in \mathbb{N}}$  that is a  $(\gamma, \beta, O(\frac{\log(1/\beta)}{(\frac{1}{2}-\gamma)^3}), O(\frac{1}{(\frac{1}{2}-\gamma)^2}))$ -ALLDC.*

► **Observation 4.2.** *If a code family  $\{C_k : \mathbb{F}_2^k \rightarrow \mathbb{F}_2^n\}_{k \in \mathbb{N}}$  is an  $(\alpha, \beta, q, L)$ -ALLDC, it is also a  $(2\alpha, \beta, 4q, 4L)$ -ALLEDC.*

Applying Observation 4.2 to Theorem 4.1, we get the ALLEDCs that we need.

► **Lemma 4.3.** *Let  $c_3 > 0$  be a constant. For every  $\gamma, \beta > 0$ , there exists a number  $f(\gamma, \beta) > 0$  and a code family  $\{C_k : \mathbb{F}_2^k \rightarrow \{0, 1\}^{f(\gamma, \beta)k^5}\}_{k \in \mathbb{N}}$  that is a  $(\gamma, \beta, \frac{c_3 \log(1/\beta)}{(1-\gamma)^3}, \frac{c_3}{(1-\gamma)^2})$ -ALLEDC.*

## 5 Strengthened Separation

In this section, we describe a property  $\mathcal{P}'$  that can be erasure-resiliently tested using a constant number of queries, but for which every tolerant tester has query complexity  $n^{\Omega(1)}$ , and prove Theorem 1.6. The following theorem implies Theorem 1.6.

- **Theorem 5.1.** *There exists a property  $\mathcal{P}'$  and constants  $\varepsilon^* \in (0, 1), c_2 > 1$  such that,*
- *For every  $\varepsilon \in \left(\frac{\varepsilon^*}{8}, 1\right)$  and  $\alpha \in \left(0, \frac{\varepsilon^*}{57600 \cdot c_2}\right)$ , property  $\mathcal{P}'$  can be  $\alpha$ -erasure-resiliently  $\varepsilon$ -tested using a constant number of queries,*
  - *For every  $\alpha \in \left(\frac{\varepsilon^*}{57600 \cdot c_2 + 2\varepsilon^*}, 1\right)$ , and  $\varepsilon' \in \left(\alpha, \frac{28800 \cdot c_2 \cdot \varepsilon^*}{28800 \cdot c_2 + \varepsilon^*}\right)$ , every  $(\alpha, \varepsilon')$ -tolerant tester for  $\mathcal{P}'$  on inputs of length  $N$  has query complexity  $N^{\Omega(1)}$ .*

## 5.1 Description of the Separating Property $\mathcal{P}'$

The property  $\mathcal{P}'$  is very similar to the property  $\mathcal{P}$  that we used in our first separation (see Definition 3.6). Like a string that satisfies  $\mathcal{P}$ , a string that satisfies  $\mathcal{P}'$  can also be thought of as consisting of a plain part (that contains the repetition of a string  $y \in \mathcal{R}_{\phi_n}$ ) and an encoded part. The encoded part of a string in  $\mathcal{P}$  is the Hadamard encoding of a string  $y \circ \pi$ , where  $\pi$  is a proof that makes the algorithm  $V$  from Claim 3.5 accept. However, the encoded part of a string satisfying  $\mathcal{P}'$  is the encoding of a string  $\pi'$ , where  $\pi'$  is a proof (whose length is asymptotically equal to  $|\pi|$ ) that makes a ‘smoothed’ PCPP accept. In addition, the encoding uses an ALLEDC (from Section 4) instead of the Hadamard code.

We first describe the ‘smoothed’ PCPP used in our construction. The following lemma by Ben-Sasson et al. [6] and Guruswami and Rudra [25] states that algorithms making nonadaptive queries can be transformed into algorithms that make nearly uniform queries.

- **Lemma 5.2** ([25, 6]). *For every nonadaptive algorithm  $T$ , there exists a mapping  $\varphi : \{0, 1\}^* \rightarrow \{0, 1\}^*$  and an algorithm  $T'$  satisfying the following:*
- *For every  $x \in \{0, 1\}^*$ , the decision of  $T$  with oracle access to  $x$  is identical to the decision of  $T'$  with oracle access to  $\varphi(x)$ . Moreover,  $3|x| < |\varphi(x)| \leq 4|x|$ , and the number of queries that  $T'$  makes to  $\varphi(x)$  is at most twice the number of queries that  $T$  makes to  $x$ .*
  - *Given oracle access to  $x' \in \{0, 1\}^n$ , each query of  $T'$  is to location  $j \in [n]$  with probability at most  $2/n$ .*

Combining Lemma 3.4 with Lemma 5.2 (along with the fact that  $\mathcal{R} = \{\mathcal{R}_{\phi_n}\}_{n \in \mathbb{N}}$  can be decided using linear-sized circuits), we get the required ‘smoothed’ PCPP for  $\mathcal{R}$ .

- **Lemma 5.3.** *Let  $c_1 > 0, c_2 > 1$  be constants. The property  $\mathcal{R}_{\phi_n}$  has a PCPP  $V$  that works for all  $\varepsilon \in (0, 1]$ , gets oracle access to an input  $y$  of length  $n$  and a proof  $\pi$  of length at most  $c_1 n \cdot \text{polylog } n$ , and makes at most  $\frac{c_2}{\varepsilon}$  queries. Moreover, the queries of  $V$  are nonadaptive and satisfy the following:*
- *Each query  $V$  makes to  $y$  is to any particular location of  $y$  with probability at most  $2/n$ .*
  - *Each query  $V$  makes to  $\pi$  is to any particular location of  $\pi$  with probability at most  $2/|\pi|$ .*

The following is the definition of our separating property  $\mathcal{P}'$ . Note that the encoded part of a string satisfying  $\mathcal{P}'$  contains the encoding of a proof as well as the complement of that encoding. This is done in order to equalize the number of 0s and 1s in the encoded part.

- **Definition 5.4** (Separating Property  $\mathcal{P}'$ ). Let  $\aleph$  and  $\varepsilon^* \in (0, 1)$  be as in Lemma 3.2. Let  $c_1 > 0, c_2 > 1$  be as in Lemma 5.3 and  $c_3 > 0$  be as in Lemma 4.3. Let  $m = \frac{28800 \cdot c_2}{\varepsilon^*}$ ,  $\gamma = \frac{1}{2} + \frac{\varepsilon^*}{57600 \cdot c_2}$  and  $\beta = \frac{\varepsilon^*}{9000c_2 \cdot \lceil \ln \frac{6c_3}{(1-\gamma)^2} \rceil}$ . For  $n \in \aleph$ , let  $p(n) \leq c_1 \cdot n \cdot \text{polylog } n$  denote the length of the proof that makes the algorithm  $V$  in Lemma 5.3 accept. Let  $f(\cdot, \cdot)$  be as in Lemma 4.3. Let  $\mathcal{C} = \{C_k\}_{k \in \mathbb{N}}$  be the  $(\gamma, \beta, \frac{c_3 \log(1/\beta)}{(1-\gamma)^3}, \frac{c_3}{(1-\gamma)^2})$ -ALLEDC from Lemma 4.3. A string  $x \in \{0, 1\}^N$  of length  $N = (m+1) \cdot 2f(\gamma, \beta) \cdot (p(n))^5$  satisfies  $\mathcal{P}'$  if the following conditions hold:

1. The first  $m \cdot 2f(\gamma, \beta) \cdot (p(n))^5$  bits of  $x$  (called the *plain part* of  $x$ ) consist of  $m \cdot \frac{2f(\gamma, \beta) \cdot (p(n))^5}{n}$  repetitions of a string  $y \in \mathcal{R}_{\phi_n}$  of length  $n$ , for  $\mathcal{R}_{\phi_n}$  from Lemma 3.2.
2. The remaining  $2f(\gamma, \beta) \cdot (p(n))^5$  bits of  $x$  is called the *encoded part*. Its first half is the encoding, using  $\mathcal{C}$ , of a string  $\pi \in \{0, 1\}^{p(n)}$  such that the PCPP  $V$  in Lemma 5.3 accepts when given oracle access to  $y \circ \pi$ . The second half of the encoded part is the complement of its first half.

## 5.2 Proof of Strengthened Separation

In this section, we prove Theorem 5.1. Lemmas 5.5 and 5.9 together imply the first and second parts of Theorem 5.1, respectively. The high level idea of the proof of Lemma 5.5 is very similar to that of Lemma 3.7. The differences arise mainly because of the way the encoded parts of strings satisfying  $\mathcal{P}$  and  $\mathcal{P}'$  differ. The erasure-resilient tester for  $\mathcal{P}$  could first check whether the plain part is a repetition of the ‘decoded input’, and then check whether the ‘decoded input’ is in  $\mathcal{R}$  with the help of the ‘decoded PCPP proof’. Since the encoded part of  $\mathcal{P}'$  is the encoding of just a PCPP proof, this is not possible. Instead, the erasure-resilient tester for  $\mathcal{P}'$  samples a uniformly random nonerased point  $u$  from the plain part and uses the ‘block’ from which  $u$  is obtained as a ‘candidate input’  $y$ . It then checks whether the plain part is a repetition of  $y$  and also checks whether  $y \in \mathcal{R}$  using the ‘approximately decoded proof’. In case a string is  $\alpha$ -erased and  $\varepsilon$ -far from  $\mathcal{P}'$ , we show that the ‘candidate input’  $y$  that we sample is  $c\alpha$ -erased and  $c'\varepsilon$ -far from  $\mathcal{R}$ , for some constants  $c, c'$ . Hence, the smoothed PCPP verifier rejects.

► **Lemma 5.5.** *Let  $\varepsilon^* \in (0, 1)$  be as in Lemma 3.2 and  $c_2 > 1$  be as in Lemma 5.3. For every  $\varepsilon \in \left(\frac{\varepsilon^*}{8}, 1\right)$  and  $\alpha \in \left(0, \frac{\varepsilon^*}{57600 \cdot c_2}\right)$ , the property  $\mathcal{P}'$  is  $\alpha$ -erasure-resiliently  $\varepsilon$ -testable using a constant number of queries.*

**Proof.** The erasure-resilient tester is presented in Algorithm 3. Let  $m$  denote  $28800 \cdot c_2 / \varepsilon^*$ . Let  $\gamma = 1/2 + \varepsilon^* / 57600 c_2$ ,  $\beta = \frac{\varepsilon^*}{9000 c_2 \cdot \lceil \ln \frac{6c_3}{(1-\gamma)^2} \rceil}$ ,  $q = \frac{c_3 \log(1/\beta)}{(1-\gamma)^3}$ , and  $L = c_3 / (1-\gamma)^2$ . For  $n \in \mathbb{N}$ , consider a string  $x \in \{0, 1\}^N$ , where  $N = (m+1) \cdot 2f(\gamma, \beta) \cdot (p(n))^5$ . The plain part of  $x$  is  $m$  times larger than the encoded part. Let  $s$  denote the number  $m \cdot 2f(\gamma, \beta) \cdot (p(n))^5 / n$ .

Assume that  $x$  satisfies  $\mathcal{P}'$ . Since  $x$  satisfies  $\mathcal{P}'$ , the plain part of  $x$  is completable to the repetitions of some  $y \in \mathcal{R}_{\phi_n}$ . Therefore, Steps 5-9 never reject. By definition of  $\mathcal{P}'$ , the first half of the encoded part of  $x$  is the encoding (using the  $(\gamma, \beta, q, L)$ -ALLED code  $\mathcal{C}$  from Lemma 4.3) of a proof  $\pi(y) \in \{0, 1\}^{p(n)}$  such that the PCPP  $V$  with oracle access to  $y \circ \pi(y)$  always accepts. The second half of the encoding is the complement of the first half. The fraction of erasures in the encoded part (even if all of the erasures were there) is at most  $(m+1)\alpha$ . Therefore, the fraction of erasures in either the first half or the second half of the encoded part is at most  $(m+1) \cdot \alpha = 1/2 + 1/2m = \gamma$ .

By the definition of a  $(\gamma, \beta, q, L)$ -ALLED code, with probability at least  $2/3$ , one of the algorithms  $T_1, T_2, \dots, T_L$  returned by the approximate local list decoder provides oracle access to  $\pi(y)$  with at most  $\beta$  fraction of errors. Let  $T_k$  be that algorithm. The tester discards this  $k$  only if an erroneous point is queried in some iteration of Steps 14-18. Since each proof query of  $V$  (in Step 17) is made to a specific index in the proof with probability at most  $2/|p(n)|$  and the string decoded by  $T_k$  is  $\beta$ -erroneous, by the union bound over queries of  $V$ , the probability of  $V$  querying an erroneous point is at most  $2\beta \cdot \frac{c_2 \cdot 75}{24\varepsilon}$ . Hence, by the union bound, the probability that the tester discards  $k$  is at most  $\frac{1}{3} + 2 \cdot 6 \cdot \lceil \ln 6L \rceil \cdot \frac{c_2 \cdot 75}{24\varepsilon} \cdot \beta \leq \frac{2}{5}$ , where the inequality follows from our setting of  $\beta$ . Hence, Step 19 does not reject with probability at least  $3/5$ . That is, the tester accepts  $x$  with probability at least  $3/5$ .



---

**Algorithm 3** Erasure-resilient tester for separating property  $\mathcal{P}'$ .
 

---

**Input:**  $\alpha, \varepsilon \in (0, 1), N = (m + 1) \cdot 2f(\gamma, \beta) \cdot (p(n))^5$ ; oracle access to  $x \in \{0, 1, \perp\}^N$

- 1: Set  $s \leftarrow m \cdot \frac{2f(\gamma, \beta) \cdot (p(n))^5}{n}$ ,  $q \leftarrow \frac{c_3 \log(1/\beta)}{(1-\gamma)^3}$ , and  $L \leftarrow \frac{c_3}{(1-\gamma)^2}$ .
- 2: Set the query budget  $Q \leftarrow 30 \cdot \left( \lceil \frac{432}{\varepsilon} \rceil \cdot \frac{2}{1-(m+1)\alpha} + L \lceil 6 \ln 6L \rceil \cdot \frac{c_2 \cdot 75}{24\varepsilon} \cdot q \right)$ .
- 3: **Accept** whenever the number of queries exceeds  $Q$ .
- 4:      $\triangleright$  Steps 5-9 check that the plain part of  $x$  is the repetition of a string  $y \in \{0, 1\}^n$ .
- 5: **repeat**  $\lceil \frac{432}{\varepsilon} \rceil$  times:
  - 6: Repeatedly sample a uniformly random point  $u$  from the plain part until  $x[u] \neq \perp$ .
  - 7: Let  $u$  be  $(i - 1)n + a$  for  $i \in [s]$  and  $a \in [n]$ .
  - 8: Repeatedly sample  $j \in [s]$  uniformly at random until  $x[(j - 1)n + a] \neq \perp$ .
  - 9: **Reject** if  $x[(i - 1)n + a] \neq x[(j - 1)n + a]$ .
- 10:  $\triangleright$  In order to query the  $i$ -th bit of the encoding, we query the  $i$ -th bits of both the first and second halves of the encoded part. We set the  $i$ -th bit of the encoding to the  $i$ -th bit of the first half if that is nonerased, and to the complement of the  $i$ -th bit of second half if that is nonerased. If both are erased, we set the  $i$ -th bit of the encoding to  $\perp$ .
- 11: Run the decoder for the  $(\gamma, \beta, q, L)$ -ALLED code (Lemma 4.3) with oracle access to the encoded part of  $x$ . Let  $A_1, A_2, \dots, A_L$  be the list of algorithms it returns.
- 12:      $\triangleright$  Steps 13-19 check that  $y \in \mathcal{R}_{\phi_n}$  using the PCPP  $V$  on decoded proofs.
- 13: **for** each  $k \in [L]$  **do**
- 14:     **repeat**  $\lceil 6 \ln 6L \rceil$  times:
  - 15: Repeatedly sample a uniformly random point  $u$  from the plain part until  $x[u] \neq \perp$ .
  - 16: Let  $u$  be  $(i - 1)n + a$  for  $i \in [s]$  and  $a \in [n]$ .
  - 17: Run the PCPP  $V$  (guaranteed by Lemma 5.3) with proximity parameter  $\frac{24\varepsilon}{75}$ , and oracle access to  $x[(i - 1)n + 1, \dots, (i - 1)n + n]$  as the input string and the string decoded by  $T_k$  as the proof.
  - 18:     **Discard** the current  $k$  if all query answers to  $V$  are nonerased and  $V$  rejects.
  - 19: **Reject** if every  $k \in [L]$  is **discarded**; otherwise, **accept**.

---

Assume now that  $x$  is  $\varepsilon$ -far from  $\mathcal{P}'$ . Let  $\mathcal{N}_{\text{pl}}$  denote the set of nonerased points in the plain part of  $x$ . Let  $\mathcal{N}_{\text{en}}$  denote the set of nonerased points in the encoded part of  $x$ . Let  $\alpha_{\text{pl}}$  denote the fraction (with respect to  $sn$ ) of erased points in the plain part. Let  $E$  denote the event that the number of queries made by the tester does not exceed the query budget  $Q$ .

► **Claim 5.6.** *The probability that Algorithm 3 exceeds its query budget is at most  $1/30$ .*

We now analyze the probability that Algorithm 3 rejects, conditioned on  $E$ .

**Case I:** the plain part of  $x$  is  $\varepsilon/144$ -far from being the repetitions of every  $y \in \{0, 1\}^n$ .

Let  $\varepsilon_{\text{pl}}$  denote the fraction of points (with respect to  $|\mathcal{N}_{\text{pl}}|$ ) in  $\mathcal{N}_{\text{pl}}$  whose values need to be changed in order to make the plain part a repetition of some string  $y \in \{0, 1\}^n$ . Let  $S_a = \{(i - 1)n + a : i \in [s]\}$  for all  $a \in [n]$ . We use the term  $a$ -th segment to refer to the set  $S_a$ . For all  $a \in [n]$ , we have  $|S_a| = s$ . For all  $a \in [n]$ , let  $\alpha_a = |\{u \in S_a : x[u] = \perp\}|/s$  denote the fraction of points in  $S_a$  that are erased. Let  $\mathcal{N}_a \subseteq S_a$  denote the set of nonerased points in the  $a$ -th segment. Let  $\varepsilon_a$  for all  $a \in [n]$  denote the fraction of points in  $\mathcal{N}_a$  whose values need to be changed in order to satisfy  $x[u] = x[v]$  for all  $u, v \in S_a$  such that both  $x[u]$  and  $x[v]$  are nonerased.



For every  $a \in [n]$  and  $u \in \mathcal{N}_a$ , the number of  $v \in \mathcal{N}_a$  such that  $x[u] \neq x[v]$ , is at least  $\varepsilon_a \cdot |\mathcal{N}_a|$ . Let  $G_a$  for all  $a \in [n]$  denote the (good) event that the tester samples a point from  $\mathcal{N}_a$  in Step 6. Let  $F$  denote the event that the tester rejects in a single iteration Steps 5-9. Hence,  $\Pr[F|E] = \sum_{a \in [n]} \Pr[G_a|E] \cdot \Pr[F|G_a, E] \geq \sum_{a \in [n]} \frac{|\mathcal{N}_a|}{|\mathcal{N}_{\text{pl}}|} \cdot \varepsilon_a = \varepsilon_{\text{pl}} \geq \varepsilon/144$ . Therefore, conditioned on  $E$ , in at least  $432/\varepsilon$  iterations, the tester will reject with probability at least  $19/20$ . Hence, the algorithm accepts with probability at most  $1/20 + \Pr[\overline{E}] \leq 1/20 + 1/30 \leq 2/5$ , since  $\Pr[\overline{E}] \leq 1/30$ . Thus, the algorithm rejects with probability at least  $3/5$ .

**Case II:** the plain part of  $x$  is  $\varepsilon/144$ -close to being repetitions of a string  $y^* \in \{0, 1\}^n$ .

► **Claim 5.7.** *The string  $y^*$  is  $\varepsilon/2$ -far from  $\mathcal{R}_{\phi_n}$ .*

Let  $B_i = \{(i-1)n + a : a \in [n]\}$  for all  $i \in [s]$ . We use the term  $i$ -th block to refer to the set  $B_i$ . For all  $i \in [s]$ , we have,  $|B_i| = n$ . Let  $\alpha_i = |\{u \in B_i : x[u] = \perp\}|/n$  for all  $i \in [s]$  denote the fraction of points in  $B_i$  that are erased. Let  $\mathcal{N}_i \subseteq S_i$  denote the set of nonerased points in the  $i$ -th block. Let  $\varepsilon_i$  for all  $i \in [s]$  denote the fraction of points in  $\mathcal{N}_i$  whose values need to be changed in order to satisfy  $x[(i-1)n + a] = y^*[a]$  for all  $a \in [n]$ .

Fix  $k \in [L]$ . We show that Algorithm 3 discards  $k$  with high probability. Consider a single iteration of Steps 15-18. Let  $y'$  denote the (partially erased) string represented by the block that Algorithm 3 samples in Step 15. Let  $G_1$  denote the (good) event that  $y'$  is  $\varepsilon/6$ -close to  $y^*$ . Let  $G_2$  denote the (good) event that  $y'$  has at most  $48\alpha$  fraction of erasures.

► **Claim 5.8.** *Conditioned on  $G_1$  and  $G_2$ , the string  $y'$  is  $24\varepsilon/75$ -far from  $\mathcal{R}_{\phi_n}$ .*

The PCPP  $V$ , with proximity parameter  $\frac{24\varepsilon}{75}$ , is run on  $y'$  and the proof decoded by  $T_k$ . Let  $B_1$  denote the (bad) event that the PCPP  $V$  obtains an erased bit as the answer to some query. Let  $B_2$  denote the (bad) event that  $V$  accepts. By Lemma 5.3, each query of  $V$  to the input part is made to each input index with probability at most  $\frac{2}{n}$  uniformly distributed among the  $n$  input indices. Hence  $\Pr[B_1|E, G_1, G_2]$ , the probability that some input query is made to an erased point, is at most  $\frac{\varepsilon_2 \cdot 75}{24\varepsilon} \cdot 96\alpha$ . The probability that the  $V$  accepts (even if there were no erased query answers) is  $\Pr[B_2|E, G_1, G_2]$  and is, by Definition 3.3, at most  $1/3$ . Thus, the probability that the PCPP accepts, conditioned on  $E$ ,  $G_1$ , and  $G_2$ , is by the union bound, at most  $\frac{\varepsilon_2 \cdot 75}{24\varepsilon} \cdot 96\alpha + \frac{1}{3} \leq \frac{1}{24} + \frac{1}{3}$ , where the inequality follows from our setting of  $\varepsilon$  and  $\alpha$ .

To bound the probability that the PCPP accepts in a single iteration of Steps 15-18, we now evaluate  $\Pr[\overline{G_1}]$  and  $\Pr[\overline{G_2}]$ . Let the random variable  $X$  denote the relative Hamming distance of  $y'$  from  $y^*$ . Then,  $\mathbb{E}[X] = \sum_{i \in [s]} \frac{|\mathcal{N}_i|}{|\mathcal{N}_{\text{pl}}|} \cdot \varepsilon_i = \varepsilon_{\text{pl}} \leq \frac{\varepsilon}{144}$ . By Markov's inequality,  $\Pr[\overline{G_1}] = \Pr[X \geq \frac{\varepsilon}{6}] \leq \mathbb{E}[X]/(\varepsilon/6) \leq 1/24$ . To bound  $\Pr[\overline{G_2}]$ , let the random variable  $Y$  denote the fraction of erasures in  $y'$ . We first show that  $\mathbb{E}[Y] = \alpha_{\text{pl}}$ . Even if all the erasures were in the plain part,  $\alpha_{\text{pl}} \leq \frac{\alpha N}{sn} \leq \alpha \cdot (1 + \frac{1}{m})$ . Again, by an application of Markov's inequality, we get  $\Pr[\overline{G_2}] = \Pr[Y > 48\alpha] \leq \frac{\mathbb{E}[Y]}{48\alpha} \leq \frac{1 + \frac{1}{m}}{48} \leq 1/24$ .

Therefore, conditioned on  $E$ , the probability that the PCPP accepts in one iteration of Steps 15-18 is at most  $\Pr[B_1|E, G_1, G_2] + \Pr[B_2|E, G_1, G_2] + \Pr[\overline{G_2}] + \Pr[\overline{G_1}] \leq \frac{1}{24} + \frac{1}{3} + \frac{1}{24} + \frac{1}{24} \leq \frac{2}{3}$ . That is, conditioned on  $E$ , for a fixed  $k \in [L]$ , in  $[6 \ln 6L]$  independent repetitions of Steps 15-18, the probability that the PCPP does not discard  $k$  is at most  $(1 - \frac{1}{3})^{[6 \ln 6L]} \leq \frac{1}{36L^2}$ . Hence, conditioned on  $E$ , the probability that for some  $k \in [L]$ , Steps 14-18 accepts is, by the union bound, at most  $1/36L$ . Thus, if  $x$  is in Case II, the probability that the tester accepts is at most,  $\frac{1}{36L} + \Pr[\overline{E}] \leq \frac{1}{36L} + \frac{1}{30} \leq \frac{2}{5}$ , where Claim 5.6 shows that  $\Pr[\overline{E}]$  is at most  $1/30$ . ◀

Next, we show that it is hard to tolerant test  $\mathcal{P}'$ . The proof of Lemma 5.9 is identical to the proof of Lemma 3.10 up to change in parameters and is hence omitted.

► **Lemma 5.9.** *Let  $\varepsilon^* \in (0, 1)$  be as in Lemma 3.2 and  $c_2 > 1$  be as in Lemma 5.3. For every  $\alpha \in (\frac{\varepsilon^*}{57600 \cdot c_2 + 2\varepsilon^*}, 1)$ , and  $\varepsilon' \in (\alpha, \frac{28800 \cdot c_2 \cdot \varepsilon^*}{28800 \cdot c_2 + \varepsilon^*})$ , every  $(\alpha, \varepsilon')$ -tolerant tester for  $\mathcal{P}'$  requires  $\tilde{\Omega}(N^{0.2})$  queries.*

**Proof of Theorem 1.6.** For sufficiently small  $\varepsilon^*$ , setting  $\alpha = \frac{\varepsilon^*}{57600 \cdot c_2 + \varepsilon^*}$  and  $\varepsilon = \frac{57599}{57600 \cdot c_2 + 2\varepsilon^*}$  satisfies all the constraints on  $\varepsilon$  and  $\alpha$  imposed by Theorem 5.1. ◀

## 6 Local Erasure-Decoding Versus Local Decoding

In this section, we prove Theorem 1.8 and an observation that if a code is locally decodable, it is also locally erasure-decodable up to (nearly) twice as many erasures. To prove Theorem 1.8, we first make the local erasure-decoder for  $\{C_n\}_{n \in \mathbb{N}}$  nonadaptive. We then show that every LEDC with a nonadaptive decoding algorithm is such that uncorrupted codewords can be locally decoded using an algorithm that queries nearly uniformly distributed codeword indices (Claim 6.2). We then use this ‘smoothness’ property (see Definition 6.1) to show that the code family is locally decodable from a smaller fraction of errors than erasures (Claim 6.3).

► **Definition 6.1** (Smooth Locally Decodable Codes). A code family  $\{C_n : \mathbb{F}_2^n \rightarrow \mathbb{F}_2^N\}_{n \in \mathbb{N}}$  is  $(q, \eta)$ -smooth locally decodable if there exists a  $(0, q)$ -local erasure-decoder  $A$  (see Definition 1.7) that, given oracle access to an uncorrupted codeword  $w \in \mathbb{F}_2^N$ , and input  $i \in [n]$ , is such that for all  $j \in [N]$ , the probability that  $A$  queries  $j$  is at most  $\eta$ .

► **Claim 6.2.** *For every  $\alpha \in [0, 1)$ , if a code  $C_n : \mathbb{F}_2^n \rightarrow \mathbb{F}_2^N$  is  $(\alpha, q)$ -locally erasure-decodable, then  $C_n$  is  $(q', \eta)$ -smooth locally decodable, where  $q' = 18q \cdot 9^{q-1}$ , and  $\eta = q'/\alpha N$ .*

► **Claim 6.3.** *For every  $\alpha \in [0, 1)$ , if a code  $C_n : \mathbb{F}_2^n \rightarrow \mathbb{F}_2^N$  is  $(q, q/\alpha N)$ -smooth locally erasure-decodable, then  $C_n$  is  $(\alpha/12q^2, 72q)$ -locally decodable.*

**Proof of Claim 6.2.** Let  $A$  be an  $(\alpha, q)$ -local erasure-decoder for  $C_n$ . We first design a nonadaptive local erasure-decoder  $A_1$  for  $C_n$  that makes a higher number of queries than  $A$ . Consider a (partially erased) codeword  $w \in (\{\perp\} \cup \mathbb{F}_2)^N$  that has at most  $\alpha$  fraction of erasures. The algorithm  $A_1$ , on oracle access to  $w$  and an input  $i \in [n]$ , has  $18 \cdot 9^{q-1}$  independent iterations. In each iteration,  $A_1$  simulates  $A$ , guesses the answers to the first  $q-1$  queries made by  $A$ , and decides the  $q$ -th query of  $A$  based on these guesses.  $A_1$  then queries  $w$  on the  $q$  queries made by  $A$  in the simulation. If all the query answers agree with the guesses, the decoder  $A_1$  stores the output of  $A$  as the output of the current iteration. Otherwise, it stores a uniformly random bit as the output of the current iteration. Finally,  $A_1$  outputs the majority value output among all iterations.

In any particular iteration, the probability that  $A_1$  outputs the correct answer in that iteration, is at least  $\frac{3^{q-1}-1}{3^q-1} \cdot \frac{1}{2} + \frac{1}{3^q-1} \cdot \frac{2}{3}$ , which is equal to  $\frac{1}{2} + \frac{1}{6 \cdot 3^{q-1}}$ . Hence, using standard arguments, the probability that  $A_1$  outputs the correct answer after  $18 \cdot 9^{q-1}$  independent iterations, is at least  $2/3$ . The query complexity of  $A_1$  is  $18q \cdot 9^{q-1}$ , which we denote by  $q'$ .

We now use  $A_1$  to construct  $A_2$ , a  $(q', \frac{q'}{\alpha N})$ -smooth local decoder for  $C_n$ . Consider an uncorrupted codeword  $w = C_n(x)$  for  $x \in \mathbb{F}_2^n$ . For each  $i \in [n]$ , let  $S_i$  denote the set consisting of indices in  $[N]$  that get queried by  $A_1$  (on input  $i$ ) with probability more than  $\frac{q'}{\alpha N}$ . Since  $\sum_{j \in [N]} \Pr[A_1^{C_n(x)}(i) \text{ queries } j] = q'$ , we have  $|S_i| \leq \alpha \cdot N$ . On input  $i \in [n]$  and oracle access to  $w = C_n(x)$ , the algorithm  $A_2$  simulates  $A_1$  in the following way. If  $A_1$  queries  $j' \in S_i$ , the algorithm  $A_2$  does not query  $j'$  and assumes that  $w[j'] = \perp$ . Thus,  $A_2$  is a  $(q', \frac{q'}{\alpha N})$ -smooth local decoder for  $C_n$ . ◀

**Proof of Claim 6.3.** Consider a  $(q, \frac{q}{\alpha N})$ -smooth local decoder  $A$  for  $C_n$ . We will construct an  $(\frac{\alpha}{12q^2}, 72q)$ -local decoder  $A'$  for  $C_n$ . Algorithm  $A'$ , on input  $i \in [n]$  and oracle access to a word  $w$  with at most  $\frac{\alpha}{12q^2}$  fraction of errors, performs 72 independent repetitions of  $A$  and outputs the majority value output among all the iterations. Let  $x \in \mathbb{F}_2^n$  be such that  $y = C_n(x)$  is the codeword closest to  $w$ . If  $A$  is run on input  $i$  with oracle access to  $y$ , then for at least  $\frac{2}{3}$  fraction of the sequences of its random coin tosses,  $A$  returns  $x_i$  correctly. When  $A$  is run on input  $i$  with oracle access to  $w$ , by the union bound and the smoothness of  $A$ , at most  $q \cdot \frac{\alpha}{12q^2} \cdot N \cdot \frac{q}{\alpha N} = \frac{1}{12}$  fraction of sequences of its random coin tosses result in an erroneous position being queried. Hence, the probability that  $A$ , on input  $i$  and oracle access to  $w$ , returns  $x_i$  correctly is at least  $\frac{2}{3} - \frac{1}{12}$ . Hence, the probability that  $A'$  outputs  $x_i$  correctly is at least  $2/3$ . The query complexity of  $A'$  is  $72q$ . ◀

► **Observation 6.4.** Every  $(\alpha, q)$ -locally decodable code  $C_n : \mathbb{F}_2^n \rightarrow \mathbb{F}_2^N$  is also  $(2\alpha - \rho, 72q)$ -locally erasure-decodable, where  $\rho = 2 \cdot \sqrt{\ln(12) \cdot \alpha/N}$ .

---

## References

- 1 Noga Alon, Jeff Edmonds, and Michael Luby. Linear Time Erasure Codes with Nearly Optimal Recovery. In *Proceedings of FOCS 1995*, pages 512–519, 1995.
- 2 László Babai, Lance Fortnow, Leonid A. Levin, and Mario Szegedy. Checking Computations in Polylogarithmic Time. In *Proceedings of STOC 1991*, pages 21–31, 1991.
- 3 Amos Beimel, Yuval Ishai, Eyal Kushilevitz, and Jean-Francois Raymond. Breaking the  $O(n^{1/(2k-1)})$  Barrier for Information-Theoretic Private Information Retrieval. In *Proceedings of FOCS 2002*, pages 261–270, 2002.
- 4 Avraham Ben-Aroya, Klim Efremenko, and Amnon Ta-Shma. A Note on Amplifying the Error-Tolerance of Locally Decodable Codes. *Electronic Colloquium on Computational Complexity (ECCC)*, 17:134, 2010.
- 5 Avraham Ben-Aroya, Klim Efremenko, and Amnon Ta-Shma. Local List Decoding with a Constant Number of Queries. In *Proceedings of FOCS 2010*, pages 715–722, 2010.
- 6 Eli Ben-Sasson, Oded Goldreich, Prahladh Harsha, Madhu Sudan, and Salil P. Vadhan. Robust PCPs of Proximity, Shorter PCPs, and Applications to Coding. *SIAM J. Comput.*, 36(4):889–974, 2006.
- 7 Eli Ben-Sasson, Prahladh Harsha, and Sofya Raskhodnikova. Some 3CNF Properties Are Hard to Test. *SIAM J. Comput.*, 35(1):1–21, 2005.
- 8 Volodia M. Blinovsky. Bounds for codes in the case of list decoding of finite volume. *Problems of Information Transmission*, 22(1):7–19, 1986.
- 9 Manuel Blum, Michael Luby, and Ronitt Rubinfeld. Self-Testing/Correcting with Applications to Numerical Problems. *J. of Computer and System Sciences*, 47(3):549–595, 1993.
- 10 Irit Dinur. The PCP theorem by gap amplification. *J. ACM*, 54(3):12, 2007.
- 11 Irit Dinur and Omer Reingold. Assignment Testers: Towards a Combinatorial Proof of the PCP Theorem. *SIAM J. Comput.*, 36(4):975–1024, 2006.
- 12 Kashyap Dixit, Sofya Raskhodnikova, Abhradeep Thakurta, and Nithin M. Varma. Erasure-Resilient Property Testing. *SIAM J. Comput.*, 47(2):295–329, 2018.
- 13 Zeev Dvir, Parikshit Gopalan, and Sergey Yekhanin. Matching Vector Codes. *SIAM J. Comput.*, 40(4):1154–1178, 2011.
- 14 Klim Efremenko. 3-Query Locally Decodable Codes of Subexponential Length. *SIAM J. on Computing*, 41(6):1694–1703, 2012.
- 15 Eldar Fischer and Lance Fortnow. Tolerant Versus Intolerant Testing for Boolean Properties. *Theory of Computing*, 2(9):173–183, 2006.

- 16 Peter Gemmell, Richard J. Lipton, Ronitt Rubinfeld, Madhu Sudan, and Avi Wigderson. Self-testing/correcting for polynomials and for approximate functions. In *Proceedings of STOC 1991*, pages 32–42, 1991.
- 17 Peter Gemmell and Madhu Sudan. Highly resilient correctors for polynomials. *Information Processing Letters*, 43(4):169–174, 1992.
- 18 Oded Goldreich, Shafi Goldwasser, and Dana Ron. Property Testing and its Connection to Learning and Approximation. *J. ACM*, 45(4):653–750, 1998.
- 19 Oded Goldreich and Leonid A. Levin. A Hard-Core Predicate for all One-Way Functions. In *Proceedings of STOC 1989*, pages 25–32, 1989.
- 20 Sivakanth Gopi, Swastik Kopparty, Rafael Mendes de Oliveira, Noga Ron-Zewi, and Shubhangi Saraf. Locally Testable and Locally Correctable Codes Approaching the Gilbert-Varshamov Bound. In *Proceedings of SODA 2017*, pages 2073–2091, 2017.
- 21 Aryeh Grinberg, Ronen Shaltiel, and Emanuele Viola. Indistinguishability by adaptive procedures with advice, and lower bounds on hardness amplification proofs. In *FOCS*, 2018. URL: <https://eccc.weizmann.ac.il/report/2018/061>.
- 22 Alan Guo and Swastik Kopparty. List-Decoding Algorithms for Lifted Codes. *IEEE Trans. Information Theory*, 62(5):2719–2725, 2016.
- 23 Venkatesan Guruswami. List decoding from erasures: bounds and code constructions. *IEEE Trans. Information Theory*, 49(11):2826–2833, 2003.
- 24 Venkatesan Guruswami and Piotr Indyk. Linear-time encodable/decodable codes with near-optimal rate. *IEEE Trans. Information Theory*, 51(10):3393–3400, 2005.
- 25 Venkatesan Guruswami and Atri Rudra. Tolerant Locally Testable Codes. In *Proceedings of RANDOM 2005*, pages 306–317, 2005.
- 26 Venkatesan Guruswami and Salil P. Vadhan. A Lower Bound on List Size for List Decoding. *IEEE Trans. Information Theory*, 56(11):5681–5688, 2010.
- 27 Dan Gutfreund and Guy N. Rothblum. The Complexity of Local List Decoding. In *Proceedings of RANDOM 2008*, pages 455–468, 2008.
- 28 Russell Impagliazzo, Ragesh Jaiswal, Valentine Kabanets, and Avi Wigderson. Uniform Direct Product Theorems: Simplified, Optimized, and Derandomized. *SIAM J. Comput.*, 39(4):1637–1665, 2010.
- 29 Swastik Kopparty. List-Decoding Multiplicity Codes. *Theory of Comput.*, 11:149–182, 2015.
- 30 Swastik Kopparty, Or Meir, Noga Ron-Zewi, and Shubhangi Saraf. High-Rate Locally Correctable and Locally Testable Codes with Sub-Polynomial Query Complexity. *J. ACM*, 64(2):11:1–11:42, 2017.
- 31 Swastik Kopparty and Shubhangi Saraf. Local List-Decoding and Testing of Random Linear Codes from High Error. *SIAM J. Comput.*, 42(3):1302–1326, 2013.
- 32 Or Meir. Combinatorial PCPs with Efficient Verifiers. *Comp. Complexity*, 23(3):355–478, 2014.
- 33 Or Meir. Combinatorial PCPs with Short Proofs. *Comp. Complexity*, 25(1):1–102, 2016.
- 34 Michal Parnas, Dana Ron, and Ronitt Rubinfeld. Tolerant property testing and distance approximation. *J. Comput. Syst. Sci.*, 72(6):1012–1042, 2006.
- 35 Alexander Polishchuk and Daniel A. Spielman. Nearly-linear size holographic proofs. In *Proceedings of STOC 1994*, pages 194–203. ACM Press, 1994.
- 36 Sofya Raskhodnikova, Noga Ron-Zewi, and Nithin Varma. Erasures versus Errors in Local Decoding and Property Testing. *Electronic Colloquium on Computational Complexity (ECCC)*, 2018. URL: <https://eccc.weizmann.ac.il/report/2018/195>.
- 37 Ronitt Rubinfeld and Madhu Sudan. Robust Characterizations of Polynomials with Applications to Program Testing. *SIAM J. Comput.*, 25(2):252–271, 1996.
- 38 Madhu Sudan, Luca Trevisan, and Salil P. Vadhan. Pseudorandom Generators without the XOR Lemma. *J. Comput. Syst. Sci.*, 62(2):236–266, 2001.

- 39 Luca Trevisan. Some Applications of Coding Theory in Computational Complexity. *CoRR*, cs.CC/0409044, 2004. [arXiv:cs/0409044](#).
- 40 Sergey Yekhanin. Towards 3-query locally decodable codes of subexponential length. *J. of the ACM*, 55(1):1:1–1:16, 2008.



# A New Approach to Multi-Party Peer-to-Peer Communication Complexity

Adi Rosén<sup>1</sup>

CNRS and Université Paris Diderot  
adiro@irif.fr

Florent Urrutia<sup>2</sup>

Université Paris Diderot  
urrutia@irif.fr

---

## Abstract

We introduce new models and new information theoretic measures for the study of communication complexity in the natural peer-to-peer, multi-party, number-in-hand setting. We prove a number of properties of our new models and measures, and then, in order to exemplify their effectiveness, we use them to prove two lower bounds. The more elaborate one is a tight lower bound of  $\Omega(kn)$  on the multi-party peer-to-peer randomized communication complexity of the  $k$ -player,  $n$ -bit function Disjointness,  $\text{Disj}_k^n$ . The other one is a tight lower bound of  $\Omega(kn)$  on the multi-party peer-to-peer randomized communication complexity of the  $k$ -player,  $n$ -bit bitwise parity function,  $\text{Par}_k^n$ . Both lower bounds hold when  $n = \Omega(k)$ . The lower bound for  $\text{Disj}_k^n$  improves over the lower bound that can be inferred from the result of Braverman et al. (FOCS 2013), which was proved in the coordinator model and can yield a lower bound of  $\Omega(kn/\log k)$  in the peer-to-peer model.

To the best of our knowledge, our lower bounds are the first tight (non-trivial) lower bounds on communication complexity in the natural *peer-to-peer* multi-party setting.

In addition to the above results for communication complexity, we also prove, using the same tools, an  $\Omega(n)$  lower bound on the number of random bits necessary for the (information theoretic) private computation of the function  $\text{Disj}_k^n$ .

**2012 ACM Subject Classification** Theory of computation  $\rightarrow$  Communication complexity, Mathematics of computing  $\rightarrow$  Information theory, Theory of computation  $\rightarrow$  Cryptographic protocols

**Keywords and phrases** communication complexity, multi-party communication complexity, peer-to-peer communication complexity, information complexity, private computation

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2019.64

**Related Version** A full version of the paper is available at <http://arxiv.org/abs/1901.01178>.

**Acknowledgements** We thank Iordanis Kerenidis and Rotem Oshman for very useful discussions.

## 1 Introduction

Communication complexity, first introduced by Yao [44], has become a major topic of research in Theoretical Computer Science, both for its own sake, and as a tool which has yielded important results (mostly lower bounds) in various theoretical computer science fields such as circuit complexity, streaming algorithms, or data structures (e.g., [34, 36, 24, 39, 23]). Communication complexity is a measure for the amount of communication needed in order to

---

<sup>1</sup> Research supported in part by ANR project RDAM.

<sup>2</sup> Research supported in part by ERC QCC and by ANR project RDAM.





solve a problem whose input is distributed among several players. The two-party case, where two players, usually called Alice and Bob, cooperate in order to compute a function of their respective inputs, has been widely studied with many important results; yet major questions in this area are still open today (e.g., the log-rank conjecture, see [34]). The multi-party case, where  $k \geq 3$  players cooperate in order to compute a function of their inputs, is much less understood.

A number of variants have been proposed in the literature to extend the two-party setting into the multi-party one. In this paper we consider the more natural *number-in-hand* (NIH) setting, where each player has its own input, as opposed to the so-called *number-on-forehead* (NOF) setting, where each player knows all pieces of the input except one, its own. Moreover, also the communication structure between the players in the multi-party setting was considered in the literature under a number of variants. For example, in the *blackboard* (or *broadcast*) model the communication between the players is achieved by each player writing, in turn, a message on the board, to be read by all other players. In the *coordinator* model, introduced in [20], there is an additional entity, the coordinator, and all players communicate back and forth only with the coordinator. The most natural setting is, however, the *peer-to-peer message-passing* model, where each pair of players is connected by a communication link, and each player can send a separate message to any other player. This latter setting has been studied, in the context of communication complexity, even less than the other multi-party settings, probably due to the difficulty in tracking the distributed communication patterns that occur during a run of a protocol in that setting. This setting is, however, not only the most natural one, and the one that occurs the most in real systems, but is the setting studied widely in the distributed algorithms and distributed computation communities, for complexity measures which are usually other than communication complexity.

In the present paper we attempt to fill this gap in the study of peer-to-peer communication complexity, and, further, to create a more solid bridge between the research field of communication complexity and the research field of distributed computation. We propose a computation model, together with an information theoretic complexity measure, for the analysis of the communication complexity of protocols in the asynchronous multi-party peer-to-peer (number-in-hand) setting. We argue that our model is, on the one hand, only a slight restriction over the asynchronous model usually used in the distributed computation literature, and, on the other hand, stronger than the models that have been previously suggested in order to study communication complexity in the peer-to-peer setting common in the distributed computation literature (e.g., [20, 42]). Furthermore, our model lends itself to the analysis of communication complexity, most notably using information theoretic tools.

Indeed, after defining our model and our information theoretic measure, that we call *Multi-party Information Cost* (MIC), we prove a number of properties of that measure, and then prove a number of fundamental properties of protocols in our model. We then exemplify the effectiveness of our model and information theoretic measure by proving two tight lower bounds. The more elaborate one is a tight lower bound of  $\Omega(kn)$ , when  $n = \Omega(k)$ , on the peer-to-peer randomized communication complexity of the function **set-disjointness** ( $\text{Disj}_k^n$ ). This function is a basic, important function, which has been the subject of a large number of studies in communication complexity, and is often seen as a test for our ability to give lower bounds in a given model (cf. [16]). We note that the communication complexity of Disjointness in the two-party case is well understood [29, 38, 3, 7, 9]. From a quantitative point of view, our result for peer-to-peer multi-party Disjointness improves by a  $\log k$  factor the lower bound that could be deduced for the peer-to-peer model from the lower bound on the communication complexity of Disjointness in the coordinator model [8]. The second lower

bound that we prove is a tight lower bound of  $\Omega(kn)$ , when  $n = \Omega(k)$ , on the peer-to-peer randomized communication complexity of the bitwise parity function  $\text{Par}_k^n$ . Both our lower bounds are obtained by giving a lower bound on the MIC of the function at hand, which yields the lower bound on the communication complexity of that function. We believe that our lower bounds are the first tight (non-trivial) lower bound on communication complexity in a peer-to-peer multi-party setting.<sup>3</sup>

It is important to note that, to the best of our knowledge, there is no known method to obtain tight lower bounds on multi-party communication complexity in a peer-to-peer setting via lower bounds in other known multi-party settings. Lower bounds obtained in the coordinator model can be transferred to the peer-to-peer model at the cost of a  $\log k$  factor, where  $k$  is the number of players, because any peer-to-peer protocol can be simulated in the coordinator model by having the players attach to every message the identity of the destination of that message [37, 21]. The loss of this factor in the lower bounds is unavoidable when the communication protocols can exploit a flexible communication pattern, since there are examples of functions where this factor in the communication complexity is necessary, while others, e.g., the parity function of single-bit inputs, have the same communication complexity in the coordinator and peer-to-peer settings (see a more detailed discussion on this point in Section 2.2). Therefore, one cannot prove tight lower bounds in the peer-to-peer setting by proving corresponding results in the coordinator model. Note that flexible communication configurations arise naturally for mobile communicating devices, for example, when these devices exchange information with the nearby devices. Constructions based on the pointer jumping problem also seem to be harder in the coordinator model, as solving the problem usually requires exchanging information in a specific order determined by the inputs of the players. It is thus important to develop lower bound techniques which apply directly in the peer-to-peer model, as we do in the present paper. Information theoretic tools seem, as we show, most suitable for this task.

**Information theoretic complexity measures.** As indicated above, our work makes use of information theoretic tools. Based on information theory, developed by Shannon [40], *Information Complexity* (IC), originally defined in [2, 14], is a powerful tool for the study of two-party communication protocols. Information complexity is a measure of how much *information*, about each other's input, the players must learn during the course of the protocol, if that protocol must compute the function correctly. Since IC can be shown to provide a lower bound on the communication complexity, this measure has proven to be a strong and useful tool for obtaining lower bounds on two-party communication complexity in a sequence of papers (e.g., [3, 4, 11, 7]). However, information complexity cannot be extended in a straightforward manner to the multi-party setting. This is because with three players or more, any function can be computed privately (cf. [5, 19]), i.e., in a way such that the players learn nothing but the value of the function to compute. This implies that the information complexity of any function is too low to provide a meaningful lower bound on the communication complexity in the natural peer-to-peer multi-party setting. Therefore, before the present paper, information complexity and its variants have been used to obtain lower bounds on multi-party communication complexity only in settings which do not allow for private protocols (and most notably not in the natural peer-to-peer setting), with the single

---

<sup>3</sup> Lower bounds in a seemingly peer-to-peer setting were given in [42]. However, in the model of that paper, the communication pattern is determined by an external view of the transcript, which makes the model equivalent to the coordinator model.

exception of [30]. For example, a number of lower bounds have been obtained via information complexity for a promise version of set-disjointness in the broadcast model [3, 13, 26] (also cf. [28]), and external information complexity was used in [10] for a lower bound on the general disjointness function, also in the broadcast model. In the coordinator model, lower bounds on the communication complexity of set-disjointness were given via variants of information complexity [8]. The latter result was extended in [15] to the function *Tribes*. A notion of external information cost in the coordinator model was introduced in [27] to study maximum matching in a distributed setting. We note that the study of communication complexity in number-in-hand multi-party settings via techniques other than those based on information theory is limited to very few papers. One such example is the technique of *symmetrization* that was introduced for the coordinator model in [37], and was shown to be useful to study functions such as the bitwise AND. That technique was further developed along with other reduction techniques in [41, 42, 43]. Another example is the notion of strong fooling sets, introduced in [12] to study deterministic communication complexity of discreet protocols, also defined in [12].

**Private computation.** It is well known that in the multi-party number-in-hand peer-to-peer setting, unlike in the two-party case, *any* function can be privately computed [5, 19]. The model that we define in the present paper does allow for (information theoretic) private computation of any function [5, 19, 1]. The minimum amount of private randomness needed in order to compute privately a given function is often referred to in this context as the *randomness complexity* of that function. Randomness complexity (in private computation) is of interest because true randomness is considered a costly resource, and since randomness complexity in private computation has been shown to be related to other complexity measures, such as the circuit size of the function or its sensitivity. For example, it has been shown [35] that a boolean function  $f$  has a linear size circuit if and only if  $f$  has constant randomness complexity. A small number of works [6, 33, 25, 30] prove lower bounds on the randomness complexity of the parity function. The parity and other modulo-sum functions are, to the best of our knowledge, the only functions for which randomness complexity lower bounds are known. Using the information theoretic results that we obtain in the present paper for the set-disjointness function, we are able to give a lower bound of  $\Omega(n)$  on the randomness complexity of  $\text{Disj}_k^n$ . The significance of this result lies in that it is the first such lower bound that grows with the size of the input (which is  $kn$ ), while the output remains a single bit, contrary to the sum function (see [6]) or the bitwise parity function (see [30]).

## 1.1 Our techniques and contributions

Our contribution in the present paper is twofold.

First, on the conceptual, modeling and definitions side we lay the foundations for proving lower bounds on (randomized) communication complexity in the natural peer-to-peer multi-party setting. Specifically, we propose a model that, on the one hand, is a very natural peer-to-peer model, and very close to the model used in the distributed computation literature, and, at the same time, does have properties that allow one to analyze protocols in terms of their information complexity and communication complexity. While at first sight the elaboration of such model does not seem to be a difficult task, many technical, as well as fundamental, issues render this task non-trivial. For example, one would like to define a notion of “transcript” that would guarantee both a relation between the length of the transcript and the communication complexity, and at the same time will contain *all* the information that the players get and use while running the protocol. The difficulty in elaborating such

model may be the reason for which, prior to the present paper, hardly any work studied communication complexity directly in a *peer-to-peer*, multi-party setting (cf. [21]), leaving the field with only the results that can be inferred from other models, hence suffering the appropriate loss in the obtained bounds. We propose our model (see Section 2.1) and prove a number of fundamental properties that allow one to analyze protocols in that model (see Section 3.2), as well as prove the accurate relationship between the entropy of the transcript and the communication complexity of the protocol (Proposition 2.4).

We then define our new information theoretic measure, that we call “Multi-party Information Cost” (MIC), intended to be applied to peer-to-peer multi-party protocols, and prove that it provides, for any (possibly randomized) protocol, a lower bound on the communication complexity of that protocol (Lemma 3.4). We further show that MIC has certain properties such as a certain direct-sum property (Theorem 3.5). We thus introduce a framework as well as tools for proving lower bounds on communication complexity in a peer-to-peer multi-party setting.

Second, we exemplify the effectiveness of our conceptual contributions by proving, using the new tools that we define, two tight lower bounds on the randomized communication complexity of certain functions in the peer-to-peer multi-party setting. Both these lower bounds are proved by giving a lower bound on the Multi-party Information Complexity of the function at hand. The more elaborate lower bound is a tight lower bound of  $\Omega(nk)$  on the randomized communication complexity of the function  $\text{Disj}_k^n$  (under the condition that  $n = \Omega(k)$ ). The function Disjointness is a well studied function in communication complexity and is often seen as a test-case of one’s ability to give lower bounds in a given model (cf. [16]). While the general structure of the proof of this lower bound does have similarities to the proof of a lower bound for Disjointness in the coordinator model [8],<sup>4</sup> we do, even in the parts that bear similarities, have to overcome a number of technical difficulties that require new ideas and new proofs. For example, the very basic *rectangularity* property of communication protocols is, in the multi-party (peer-to-peer) setting, very sensitive to the details of the definition of the model and the notion of a transcript. We therefore need first to give a proof of this property in the peer-to-peer model (Lemma 3.6 and Lemma 3.7). We then use a distribution of the input which is a modification over the distributions used in [8, 15] (see Section 5). Our proof proceeds, as in [8], by proving a lower bound for the function AND, on a certain information theoretic measure that, in our proof, is called SMIC (for Switched Multi-party Information Cost), and then, by using a direct-sum-like lemma, to infer a lower bound on SMIC for Disjointness (we note that SMIC is an adaptation to the peer-to-peer model of a similar measure used in [8]). However, the lack of a “coordinator” in a peer-to-peer setting necessitates a definition of a more elaborate reduction protocol, and a more complicated proof for the direct-sum argument, inspired by classic secret-sharing primitives. See Lemma 6.1 for our construction and proof. We then show that SMIC provides a lower bound on MIC, which yields our lower bound on the communication complexity of Disjointness.

We further give a tight lower bound of  $\Omega(nk)$  on the randomized communication complexity of the function  $\text{Par}_k^n$  (bitwise parity) in the peer-to-peer multi-party setting (under the condition that  $n = \Omega(k)$ ). This proof proceeds by first giving a lower bound on MIC for the parity function  $\text{Par}_k^1$ , and then using a direct-sum property of MIC to get a lower bound on MIC for  $\text{Par}_k^n$ . The latter yields the lower bound of  $\Omega(nk)$  on the communication complexity of  $\text{Par}_k^n$ .

<sup>4</sup> The lower bound in [8] would yield an  $\Omega(\frac{1}{\log k} \cdot nk)$  lower bound in the peer-to-peer setting.

To the best of our knowledge, our lower bounds are the first tight (non-trivial) lower bound on communication complexity in a peer-to-peer multi-party setting.

In addition to our results on communication complexity, we analyze the number of random bits necessary for private computations [5, 19], making use of the model, tools and techniques we develop in the present paper. It has been shown [30] that the *public information cost* (defined also in [30]) can be used to derive a lower bound on the randomness complexity of private computations. In the present paper we give a lower bound on the public information cost of any synchronous protocol computing the Disjointness function by relating it to its Switched Multi-party Information Cost, which yields the lower bound on the randomness complexity of Disjointness.

**Organization.** Due to space limitation *all* proofs are deferred to the full version of the paper. Section 2 introduces our model. In Section 3 we define our new information theoretic measure, MIC, give some of its properties, and give a number of fundamental properties of protocols in our model. In Section 4 we give the lower bound for the bitwise parity function. In Section 5 we prove a lower bound on the switched multi-party information cost of  $\text{AND}_k$ , and in Section 6, we prove, using the results of Section 5, the lower bound on the communication complexity of  $\text{Disj}_k^n$ . In Section 7 we apply our information theoretic lower bounds in order to give a lower bound on the number of random bits necessary for the private computation of  $\text{Disj}_k^n$ . Last, in Section 8 we discuss some open questions.

## 2 Multi-party communication protocols

We start with our model, and, to this end, give a number of notations.

**Notations.** We denote by  $k$  the number of players. We often use  $n$  to denote the size (in bits) of the input to each player. Calligraphic letters will be used to denote sets. Upper case letters will be used to denote random variables, and given two random variables  $A$  and  $B$ , we will denote by  $AB$  the joint random variable  $(A, B)$ . Given a string (of bits)  $s$ ,  $|s|$  denotes the length of  $s$ . Using parentheses we denote an ordered set (family) of items, e.g.,  $(Y_i)$ . Given a family  $(Y_i)$ ,  $Y_{-i}$  denotes the sub-family which is the family  $(Y_i)$  *without* the element  $Y_i$ . The letter  $X$  will usually denote the input to the players, and we thus use the shortened notation  $X$  for  $(X_i)$ , i.e., the input to all players. A protocol will usually be denoted by  $\pi$ .

We now define a natural communication model which is a slight restriction of the general asynchronous peer-to-peer model. The restriction of our model compared to the general asynchronous peer-to-peer model is that for a given player at a given time, the set of players from which that player waits for a message before sending any message of its own is determined by that player's own local view, i.e., from that player's input and the messages it has read so far, as well as its private randomness, and the public randomness. This allows us to define information theoretic tools that pertain to the transcripts of the protocols, and at the same time to use these tools as lower bounds for communication complexity. This restriction however does not exclude the existence of private protocols, as other special cases of the general asynchronous model do. We observe that practically all multi-party protocols in the literature are implicitly defined in our model, and that without such restriction, one bit of communication can bring  $\log k$  bits of information, because not only the content of the message, but also the identity of the sender may reveal information. To exemplify why the general asynchronous model is problematic consider the following simple example (that we borrow from our work in [30]).

► **Example 2.1.** There are 4 players  $A$ ,  $B$  and  $C$ ,  $D$ . The protocol allows  $A$  to transmit to  $B$  its input bit  $x$ . But all messages sent in the protocol are the bit 0, and the protocol generates only a single transcript over all possible inputs. The protocol works as follows:

**A:** If  $x = 0$  send 0 to  $C$ ; after receiving 0 from  $C$ , send 0 to  $D$ .

If  $x = 1$  send 0 to  $D$ ; after receiving 0 from  $D$ , send 0 to  $C$

**B:** After receiving 0 from a player, send 0 back to that player.

**C,D:** After receiving 0 from  $A$  send 0 to  $B$ . After receiving 0 from  $B$  send 0 to  $A$ .

It is easy to see that  $B$  learns the value of  $x$  from the order of the messages it gets.

In what follows we formally define our model, compare it to the general one and to other restricted ones, and explain the usefulness and logic of our specific model.

## 2.1 Definition of the model

We work in a *multi-party, number-in-hand, peer-to-peer* setting. Each player  $1 \leq i \leq k$  has unbounded local computation power and, in addition to its input  $X_i$ , has access to a source of private randomness  $R_i$ . We will use the notation  $R$  for  $(R_i)$ , i.e., the private randomness of all players. A source of public randomness  $R^p$  is also available to all players. We will call a protocol with no private randomness a public-coins protocol. The system consists of  $k$  players and a family of  $k$  functions  $f = (f_i)_{i \in [1, k]}$ , with  $\forall i \in [1, k]$ ,  $f_i : \prod_{\ell=1}^k \mathcal{X}_\ell \rightarrow \mathcal{Y}_i$ , where  $\mathcal{X}_\ell$  denotes the set of possible inputs of player  $\ell$ , and  $\mathcal{Y}_i$  denotes the set of possible outputs of player  $i$ . The players are given some input  $x = (x_i) \in \prod_{i=1}^k \mathcal{X}_i$ , and for every  $i$ , player  $i$  has to compute  $f_i(x)$ .

We define the communication model as follows, which is the asynchronous setting, with some restrictions. To make the discussion simpler we assume a global time which is *unknown* to the players. Every pair of players is connected by a bidirectional communication link that allows them to send messages to each other. There is no bound on the delivery time of a message, but every message is delivered in finite time, and the communication link maintains FIFO order in each of the two directions. Given a specific time we define the *view* of player  $i$  as the input of this player,  $X_i$ , its private randomness,  $R_i$ , the public randomness,  $R^p$ , and the messages read so far by player  $i$ . After the protocol has started, each player runs the protocol in *local rounds*. In each round, player  $i$  sends messages to some subset of the other players. The identity of these players, as well as the content of these messages, depend on the current view of player  $i$ . The player also decides whether it should stop, and output (or “return”) the result of the function  $f_i$ . Then (if player  $i$  did not stop and return the output), the player waits for messages from a certain subset of the other players, this subset being also determined by the current view of the player. Then the (local) round of player  $i$  terminates.<sup>5</sup> To make it possible for the player to identify the arrival of the *complete* message that it waits for, we require that each message sent by a player in the protocol is self-delimiting.

Denote by  $\mathcal{D}_i^\ell$  the set of possible views of player  $i$  at the end of local round  $\ell$ ,  $\ell \geq 0$ , where the beginning of the protocol is considered round 0.

Formally, a **protocol**  $\pi$  is defined by a set of local programs, one for each player  $i$ , where the local program of player  $i$  is defined by a sequence of functions, parametrized by the index of the *local* round  $\ell$ ,  $\ell \geq 1$ :

<sup>5</sup> The fact that the receiving of the incoming messages comes as the last step of the (local) round comes only to emphasize that the sending of the messages and the output are a function of only the messages received in previous (local) rounds.

- $S_i^{\ell,s} : \mathcal{D}_i^{\ell-1} \rightarrow 2^{\{1,\dots,k\}\setminus\{i\}}$ , defining the set of players to which player  $i$  sends the messages.
- $m_{i,j}^\ell : \mathcal{D}_i^{\ell-1} \rightarrow \{0,1\}^*$ , such that for any  $D_i^{\ell-1} \in \mathcal{D}_i^{\ell-1}$ , if  $j \in S_i^{\ell,s}(D_i^{\ell-1})$ , then  $m_{i,j}^\ell(D_i^{\ell-1})$  is the content of the message player  $i$  sends to player  $j$ . Each such message is self-delimiting.
- $O_i^\ell : \mathcal{D}_i^{\ell-1} \rightarrow \{0,1\}^* \cup \{\perp\}$ , defining whether or not the local program of player  $i$  stops and the player returns its output, and what is that output. If the value is  $\perp$  then no output occurs. If the value is  $y \in \{0,1\}^*$ , then the local program stops and the player returns the value  $y$ .
- $S_i^{\ell,r} : \mathcal{D}_i^{\ell-1} \rightarrow 2^{\{1,\dots,k\}\setminus\{i\}}$ , defining the set of players from which player  $i$  waits to receive a message.

To define the **transcript** of a protocol we proceed as follows. We first define  $k(k-1)$  basic transcripts  $\Pi_{i,j}^r$ , denoting the transcript of the messages read by player  $i$  from its link from player  $j$ , and another  $k(k-1)$  basic transcripts  $\Pi_{i,j}^s$ , denoting the transcript of the messages sent by player  $i$  on its link to player  $j$ .

We then define the transcript of player  $i$ ,  $\Pi_i$ , as the  $2(k-1)$ -tuple of the  $2(k-1)$  basic transcripts  $\Pi_{i,j}^r, \Pi_{i,j}^s, j \in [1, k] \setminus \{i\}$ . The transcript of the whole protocol  $\Pi$  is defined as the  $k$ -tuple of the  $k$  player transcripts  $\Pi_i, i \in [1, k]$ . We denote by  $\Pi_i(x, r)$  the transcript of player  $i$  when protocol  $\pi$  is run on input  $x$  and on randomness (public and private of all players)  $r$ . By  $\Pi_i^\ell(x, r)$  we denote  $\Pi_i(x, r)$  modified such that all the messages that player  $i$  sends in local rounds  $\ell' > \ell$ , and all the messages that player  $i$  reads in local rounds  $\ell' > \ell$  are eliminated from the transcript. Observe that while  $\Pi_{i,j}^r$  is always a prefix of  $\Pi_{j,i}^s$ , the definition of a protocol does not imply that they are equal. Further observe that each bit sent in  $\pi$  appears in  $\Pi$  at most twice.

We note that while seemingly the model that we introduce here is the same as the one used in [30], there are important differences between the models, and that these differences are crucial for the properties that we prove in the present paper to hold. See Section 2.2 for a comparison.

For a  $k$ -party protocol  $\pi$  we denote the set of possible inputs as  $\mathcal{X}$ , and denote the projection of this set on the  $i$ 'th coordinate (i.e., the set of possible inputs for player  $i$ ) by  $\mathcal{X}_i$ . Thus  $\mathcal{X} \subseteq \mathcal{X}_1 \times \dots \times \mathcal{X}_k$ . The set of possible transcripts for a protocol is denoted  $\mathcal{T}$ , and the projection of this set on the  $i$ 'th coordinate (i.e., the set of possible transcripts of player  $i$ ) is denoted  $\mathcal{T}_i$ . Observe that  $\mathcal{T} \subseteq \mathcal{T}_1 \times \dots \times \mathcal{T}_k$ .

Furthermore, in the course of the proofs, we sometimes consider a protocol that does *not* have access to public randomness (but may have private randomness). We call such protocol a *private-coins protocol*.

We now formally define the notion of a protocol computing a given function with certain bounded error. We will give most of the following definitions for the case where all functions  $f_i$  are the same function, that we denote by  $f$ . The definitions in the case of family of functions are similar.

► **Definition 2.2.** For a given  $0 \leq \epsilon < 1$ , a protocol  $\pi$   $\epsilon$ -computes a function  $f$  if for all  $x \in \Pi_{i=1}^k \mathcal{X}_i$ :

- For all possible assignments for the random sources  $R_i, 1 \leq i \leq k$ , and  $R^p$ , every player eventually stops and returns an output.
- With probability at least  $1 - \epsilon$  (over all random sources) the following event occurs: each player  $i$  outputs the value  $f(x)$ , i.e., the correct value of the function.

We also consider the notion of *external computation*.



► **Definition 2.3.** For a given  $0 \leq \epsilon < 1$ , a protocol  $\pi$  *externally*  $\epsilon$ -computes  $f$  if there exists a deterministic function  $\theta$  taking as input the possible transcripts of  $\pi$  and verifying  $\forall x \in \mathcal{X}$ ,  $\Pr[\theta(\Pi(x)) = f(x)] \geq 1 - \epsilon$ .

The **communication complexity** of a protocol is defined as the worst case, over the possible inputs and the possible randomness, of the number of bits sent by all players. For a protocol  $\pi$  we denote its communication complexity by  $\text{CC}(\pi)$ . For a given function  $f$  and a given  $0 \leq \epsilon < 1$ , we denote by  $\text{CC}^\epsilon(f)$  the  $\epsilon$ -error communication complexity of  $f$ , i.e.,  $\text{CC}^\epsilon(f) = \inf_{\pi \text{ } \epsilon\text{-computing } f} \text{CC}(\pi)$ .

Finally, we give a proposition that relates the communication complexity of a  $k$ -party protocol  $\pi$  to the entropy of the transcripts of the protocol  $\pi$ .

► **Proposition 2.4.** *Let the input to a  $k$ -party protocol  $\pi$  be distributed according to an arbitrary distribution. Then,  $\sum_{i=1}^k H(\Pi_i) \leq 4 \cdot \text{CC}(\pi) + 4k^2$ , where the entropy is according to the input distribution and the randomization of protocol  $\pi$ .*

## 2.2 Comparison to other models

The somewhat restricted model (compared to the general asynchronous model) that we work with allows us to use information theoretic tools for the study of protocols in this model, and in particular to give lower bounds on the multi-party communication complexity. Notice that the general asynchronous model is problematic in this respect since one bit of communication can bring  $\log k$  bits of information, because not only the content of the message, but also the identity of the sender may reveal information. Thus, information cannot be used as a lower bound on communication. In our case, the sets  $S_i^{l,r}$  and  $S_i^{l,s}$  are determined by the current view of the player,  $\Pi$  contains only the content of the messages, and thus the desirable relation between the communication and the information is maintained. On the other hand, our restriction is natural, does not seem to be very restrictive (practically all protocols in the literature adhere to our model), and does not exclude the existence of private protocols. To exemplify why the general asynchronous model is problematic see Example 2.1.

While the model that we introduce in the present paper bears some similarities to the model used in [30], there are a number of important differences between them. First, the definition of the transcript is different, resulting in a different relation between the entropy of the transcript and the communication complexity. More important is the natural property of the model in the present paper that the local program of a protocol in a given node ends its execution when it locally gives its output. It turns out that the very basic rectangularity property of protocols, used in many papers, holds in this case (and when the transcript is defined as we define in the present paper), while if the local protocol may continue to operate after output, there are examples where this property does not hold. Thus, we view the introduction of the present model also as a contribution towards identifying the necessary features of a peer-to-peer model so that basic and useful properties of protocols hold in the peer-to-peer setting.

There has been a long series of works about multi-party communication protocols in different variants of models, for example [20, 13, 26, 28, 37, 17, 18] (see [21] for a comparison of a few of these models). In the *coordinator model* (cf. [20, 37, 8]), an additional player (the coordinator) with no input can communicate privately with each player, and the players can only communicate with the coordinator. We first note that the coordinator model does not yield exact bounds for the multi-party communication complexity in the peer-to-peer setting (neither in our model nor in the most general one). Namely, any protocol in the peer-to-peer model can be transformed into a protocol in the coordinator model with an

$O(\log k)$  multiplicative factor in the communication complexity, by sending each message to the coordinator with an  $O(\log k)$ -bit label indicating its destination. This factor is sometimes necessary, e.g., for the **permutation** functional defined as follows: Given a permutation  $\sigma : \llbracket 1, k \rrbracket \rightarrow \llbracket 1, k \rrbracket$ , each player  $i$  has as input a bit  $b_i$  and  $\sigma^{-1}(\sigma(i) - 1)$  and  $\sigma^{-1}(\sigma(i) + 1)$  (i.e., each player has as input the indexes of the players before and after itself in the permutation).<sup>6</sup> For player  $i$  the function  $f_i$  is defined as  $f_i = b_{\sigma^{-1}(\sigma(i)+1)}$  (i.e., the value of the input bit of the next player in the permutation  $\sigma$ ). Clearly in our model the communication complexity of this function is  $k$  (each player sends its input bit to the correct player), and the natural protocol is valid in our model. On the other hand, in the coordinator model  $\Omega(k \log k)$  bits of communication are necessary. But this multiplicative factor between the complexities in the two models is not always necessary: the communication complexity of the parity function **Par** is  $\Theta(k)$  both in the peer-to-peer model and in the coordinator model.

Moreover, when studying private protocols in the multi-party setting, the coordinator model does not offer any insight. In the coordinator model, described in [20] and used for instance in [8], if one does not impose any privacy requirement with respect to the coordinator, it is trivial to have a private protocol by all players sending their input to the coordinator, and the coordinator returning the results to the players. If there is a privacy requirement with respect to the coordinator, then if there is a random source shared by all the players (but not the coordinator), privacy is always possible using the protocol of [22]. If no such source exists, privacy is impossible in general. This follows from the results of Braverman et al. [8] who show a non-zero lower bound on the total internal information complexity of all parties (including the coordinator) for the function *Disjointness* in that model. Our model, on the other hand, does allow for the private computation of any function [5, 19, 1].

It is worthwhile to contrast our model, and the communication complexity measure that we are concerned with, with work in the so-call congested-clique model that has gained increasing attention in the distributed computation literature (cf. [31, 32]). While both models are based on a communication network in the form of a complete graph (i.e., every player can send messages to any other player, and these messages can be different) there are two significant differences between them. Most of the works in the congested clique model deal with graph-theoretic problems and the input to each player is related to the adjacency list of a node (identified with that player) in the input graph, while in our model the input is not associated in any way with the communication graph. More importantly, the congested clique model is a *synchronous* model while ours is an asynchronous one. This brings about a major difference between the complexity measures studied in each of the models. Work in the congested clique model is concerned with giving bounds on the number of rounds necessary to fulfill a certain task under the condition that in each round each player can send to any other player a limited number of bits (usually  $O(\log k)$  bits). The measure of communication complexity, that is of interest to us in the present paper, deals with the total number of communication bits necessary to fulfill a certain task in an asynchronous setting without any notion of global rounds.<sup>7</sup>

<sup>6</sup> All additions are modulo  $k$ . This is a promise problem.

<sup>7</sup> Any function can be computed in the congested clique model with  $O(k)$  communication complexity (at a cost of having many rounds) by each player, having input  $x$ , sending a single bit to player 1 only at round number  $x$ . On the other hand, in the asynchronous model any function can be computed in a single “round” (at a cost of high communication complexity) by each player sending its whole input to player 1.

### 3 Tools for the study of multi-party communication protocols

In this section we consider two important tools for the study of peer-to-peer multi-party communication protocols. First, we define and introduce an information theoretic measure that we call *Multi-party Information Cost* (MIC); we later use it to prove our lower bounds. Then, we prove, in the peer-to-peer multi-party model that we define, the so-called rectangularity property of communication protocols, that we also use in our proofs.

#### 3.1 Multi-party Information Cost

We now introduce an information theoretic measure for multi-party peer-to-peer protocols that we later show to be useful for proving lower bounds on the communication complexity of multi-party peer-to-peer protocols. We note that a somewhat similar measure was proposed in [8] for the coordinator model, but, to the best of our knowledge, never found an application as a tool in a proof of a lower bound.

► **Definition 3.1.** For any  $k$ -player protocol  $\pi$  and any input distribution  $\mu$ , we define the *multi-party information cost* of  $\pi$ :

$$\text{MIC}_\mu(\pi) = \sum_{i=1}^k (I(X_{-i}; \Pi_i | X_i R_i) + I(X_i; \Pi_i | X_{-i} R_{-i})) .$$

Observe that the second part of each of the  $k$  summands can be interpreted as the information that player  $i$  “leaks” to the other players on its input. While the “usual” intuitive interpretation of two-party IC is “what Alice learns on Bob’s input plus what Bob learns on Alice’s input”, one can also interpret two-party IC as “what Alice learns on Bob’s input plus what Alice leaks on her input”. Thus, MIC can be interpreted as summing over all players  $i$  of “what player  $i$  learns on the other players’ inputs, plus what player  $i$  leaks on its input.” Indeed, the expression defining MIC is equal to the sum, over all players  $i$ , of the two-party IC for the two-party protocol that results from collapsing all players, except  $i$ , into one virtual player. Thus, for number of players  $k = 2$ ,  $\text{MIC} = 2 \cdot \text{IC}$ . We note that defining our measure without the private randomness in the condition of the mutual information expressions would yield the exact same measure (as is the case for 2-party IC); we prefer however to define MIC with the randomness in the conditions, as we believe that it allows one to give shorter, but still clear and accurate, proofs.

On the other hand observe that the second of the two mutual information expressions has  $X_{-i}$  in the condition, contrary to a seemingly similar measure used in [8] (Definition 3 in [8]). Our measure is thus “internal” in nature, while the one of [8] has an “external” component. The fact that MIC is “internal” allows us to give lower bounds on MIC, and thus to use it for lower bounds on the communication complexity, contrary to the measure of [8].

Further observe that the summation, over all players, of each one of the two mutual information expressions alone would not yield a measure useful for proving lower bounds on the communication complexity of functions. The first mutual information expression would yield a measure for functions that would never be higher than the entropy of the function at hand, due to the existence of private protocols for all functions [5, 19]. For the second mutual information expression there are functions for which that measure would be far too low compared to the communication complexity: e.g., the function  $f = x_1$ ,  $x \in \{0, 1\}^n$  (i.e., the value of the function is the input of player 1); in that case the measure would equal only  $n$ , while the communication complexity of that function is  $\Omega(kn)$ .

We now define the multi-party information complexity of a function.

► **Definition 3.2.** For any function  $f$ , any input distribution  $\mu$ , and any  $0 \leq \epsilon \leq 1$ , we define the quantity

$$\text{MIC}_\mu^\epsilon(f) = \inf_{\pi \text{ } \epsilon\text{-computing } f} \text{MIC}_\mu(\pi) .$$

► **Definition 3.3.** For any  $f$ , and any  $0 \leq \epsilon \leq 1$ , we define the quantity

$$\text{MIC}^\epsilon(f) = \inf_{\pi \text{ } \epsilon\text{-computing } f} \sup_{\mu} \text{MIC}_\mu(\pi) .$$

We now claim that the multi-party information cost and the communication complexity of a protocol are related, as formalized by the following lemma.

► **Lemma 3.4.** For any  $k$ -player protocol  $\pi$ , and for any input distribution  $\mu$ ,

$$\text{CC}(\pi) \geq \frac{1}{8} \text{MIC}_\mu(\pi) - k^2 .$$

We now show that the multi-party information cost satisfies a direct sum property for product distributions. In what follows, the notation  $f^{\otimes n}$  denotes the task of computing  $n$  instances of  $f$ , where the requirement from an  $\epsilon$ -computing protocol is that each instance is computed correctly with probability at least  $1 - \epsilon$  (as opposed to the stronger requirement that the whole vector of instances is computed correctly with probability at least  $1 - \epsilon$ ).

► **Theorem 3.5.** For any protocol  $\pi$  (externally)  $\epsilon$ -computing a function  $f^{\otimes n}$ , there exists a protocol  $\pi'$  (externally)  $\epsilon$ -computing  $f$  such that, for any product distribution  $\mu$  for the input, it holds that

$$\text{MIC}_{\mu^n}(\pi) \geq n \cdot \text{MIC}_\mu(\pi') .$$

## 3.2 The rectangularity property

The *rectangularity property* (or *Markov property*) is one of the key properties that follow from the structure and definition of (some) protocols. For randomized protocols it was introduced in the two-party setting and in the multi-party blackboard model in [3], and in the coordinator model in [8]. We prove a similar rectangularity property in the peer-to-peer model that we consider in the present paper.

We note that the proof of this property in the peer-to-peer model makes explicit use of the specific properties of the model we defined: the proof that follows explicitly uses the definition of the transcript on an edge by edge basis as in our model, as well as the fact that a player returns and stops as one operation. One can build examples where if any of these two properties does not hold, then the rectangularity property of protocols does not hold. Thus we view the following proof of rectangularity in our model also as an identification of model properties needed for the useful rectangularity property of multiparty peer-to-peer protocols to hold.

To define this property, for any transcript  $\bar{\tau} \in \mathcal{T}_i$ , let  $\mathcal{A}_i(\bar{\tau}) = \{(x, r) \mid \Pi_i(x, r) = \bar{\tau}\}$  (i.e., the set of input, randomness pairs that lead to transcript  $\bar{\tau}$ ), and define the projection of  $\mathcal{A}_i(\bar{\tau})$  on coordinate  $i$  as  $\mathcal{I}_i(\bar{\tau}) = \{(x', r'), \exists (x, r) \in \mathcal{A}_i(\bar{\tau}), x' = x_i \ \& \ r' = r_i\}$ , and the projection of  $\mathcal{A}_i(\bar{\tau})$  on the complement of coordinate  $i$  as  $\mathcal{J}_i(\bar{\tau}) = \{(x', r'), \exists (x, r) \in \mathcal{A}_i(\bar{\tau}), x' = x_{-i} \ \& \ r' = r_{-i}\}$ . Similarly, for any transcript  $\tau \in \mathcal{T}$ , let  $\mathcal{B}(\tau) = \{(x, r) \mid \Pi(x, r) = \tau\}$ , and for any player  $i$ , let  $\mathcal{H}_i(\tau) = \{(x', r'), \exists (x, r) \in \mathcal{B}(\tau), x' = x_{-i} \ \& \ r' = r_{-i}\}$ .

We start by proving a combinatorial property of transcripts of communication protocols, which intuitively follows from the fact that each player has access to only its own input and private randomness. The proof of this property is technically more involved compared to the analogous property in other settings, since the structure of protocols and the manifestation of the transcripts in the peer-to-peer setting are more flexible than in the other settings.

► **Lemma 3.6.** *Let  $\pi$  be a  $k$ -player private-coins protocol with inputs from  $\mathcal{X} = \mathcal{X}_1 \times \dots \times \mathcal{X}_k$ . Let  $\mathcal{T}$  denote the set of possible transcripts of  $\pi$ , and for  $i \in \llbracket 1, k \rrbracket$  let  $\mathcal{T}_i$  denote the set of possible transcript observed by player  $i$ , so that  $\mathcal{T} \subseteq \mathcal{T}_1 \times \dots \times \mathcal{T}_k$ . Then,  $\forall i \in \llbracket 1, k \rrbracket$ :*

- $\forall \bar{\tau} \in \mathcal{T}_i, \mathcal{A}_i(\bar{\tau}) = \mathcal{I}_i(\bar{\tau}) \times \mathcal{J}_i(\bar{\tau})$ .
- $\forall \tau \in \mathcal{T}, \mathcal{B}(\tau) = \mathcal{I}_i(\tau_i) \times \mathcal{H}_i(\tau)$ .

We now prove the *rectangularity property of randomized protocols* in the peer-to-peer setting.

► **Lemma 3.7.** *Let  $\pi$  be a  $k$ -player private-coins protocol with inputs from  $\mathcal{X} = \mathcal{X}_1 \times \dots \times \mathcal{X}_k$ . Let  $\mathcal{T}$  denote the set of possible transcripts of  $\pi$ , and for  $i \in \llbracket 1, k \rrbracket$  let  $\mathcal{T}_i$  denote the set of possible transcript observed by player  $i$ , so that  $\mathcal{T} \subseteq \mathcal{T}_1 \times \dots \times \mathcal{T}_k$ . Then for every  $i \in \llbracket 1, k \rrbracket$ , there exist functions  $q_i : \mathcal{X}_i \times \mathcal{T}_i \rightarrow [0, 1]$ ,  $q_{-i} : \mathcal{X}_{-i} \times \mathcal{T}_i \rightarrow [0, 1]$  and  $p_{-i} : \mathcal{X}_{-i} \times \mathcal{T} \rightarrow [0, 1]$  such that  $\forall x \in \mathcal{X}, \forall \tau = (\tau_1, \dots, \tau_k) \in \mathcal{T}, \Pr[\Pi_i(x) = \tau_i] = q_i(x_i, \tau_i)q_{-i}(x_{-i}, \tau_i)$ , and  $\forall x \in \mathcal{X}, \forall \tau = (\tau_1, \dots, \tau_k) \in \mathcal{T}, \Pr[\Pi(x) = \tau] = q_i(x_i, \tau_i)p_{-i}(x_{-i}, \tau)$ .*

## 4 The function parity

We now prove a lower bound on the multi-party peer-to-peer randomized communication complexity of the  $k$ -party  $n$ -bit parity function  $\text{Par}_k^n$ , defined as follows: each player  $i$  receives  $n$  bits  $(x_i^p)_{p \in \llbracket 1, n \rrbracket}$  and player 1 has to output the bitwise sum modulo 2 of the inputs, i.e.,  $\text{Par}_k^n(x) = (\oplus_{i=1}^k x_i^1, \oplus_{i=1}^k x_i^2, \dots, \oplus_{i=1}^k x_i^n)$  (the case where all  $k$  players compute the function is trivial). To start, we prove a lower bound on the multi-party information complexity of the parity function, where each player has a single input bit. For simplicity we denote this function  $\text{Par}_k$ , rather than  $\text{Par}_k^1$ .

► **Theorem 4.1.** *Let  $\mu$  be the uniform distribution on  $\{0, 1\}^k$ . Given any fixed  $0 \leq \epsilon < \frac{1}{2}$ , for any protocol  $\pi$   $\epsilon$ -computing  $\text{Par}_k$ , it holds that  $\text{MIC}_\mu(\pi) = \Omega(k)$ .*

The next theorem follows immediately from Theorem 4.1 and Theorem 3.5.

► **Theorem 4.2.** *Let  $\mu$  be the uniform distribution on  $\{0, 1\}^k$ . Given any fixed  $0 \leq \epsilon < \frac{1}{2}$ , for any protocol  $\pi$   $\epsilon$ -computing  $\text{Par}_k^n$ , it holds that  $\text{MIC}_{\mu^n}(\pi) = \Omega(kn)$ .*

We can now prove a lower bound on the communication complexity of  $\text{Par}_k^n$ . Note that the lower bound for  $\text{Par}_k^n$  given in [30] is valid only for a restricted class of protocols, called “oblivious” in [30].

► **Theorem 4.3.** *Given any fixed  $0 \leq \epsilon < \frac{1}{2}$ , there is a constant  $\alpha$  such that for  $n \geq \frac{1}{\alpha}k$ ,*

$$\text{CC}^\epsilon(\text{Par}_k^n) = \Omega(kn) .$$

## 5 The function AND

In this section we consider an arbitrary  $k$ -party protocol,  $\pi$ , where each player has an input bit  $x_i$ , and where  $\pi$  has to compute the AND of all the input bits. We prove a lower bound on a certain information theoretic measure (that we define below) for  $\pi$ . The proof makes use of the following input distribution.

**Input distribution.** Consider the distribution  $\mu$  defined as follows. Draw a bit  $M \sim \text{Ber}(\frac{2}{3}, \frac{1}{3})$ , and a uniformly random index  $Z \in \llbracket 1, k \rrbracket$ . Assign 0 to  $X_Z$ . If  $M = 0$ , sample  $X_{-Z}$  uniformly in  $\{0, 1\}^{k-1}$ ; if  $M = 1$ , assign  $1^{k-1}$  to  $X_{-Z}$ . We will also work with the product distribution  $\mu^n$ . Our distribution is similar to the ones of [8, 15] in that it leads to a high information cost (or similar measures) for the function  $\text{AND}_k$ . The distribution that we use has the property that the AND of any input in the support of  $\mu$  is 0. This allows us to prove lower bounds for the Disjointness function without the constraint that  $k = \Omega(\log n)$  which was necessary in [8] (but not in [15]).

### 5.1 Switched multi-party information cost of $\text{AND}_k$

We propose the following definition, which is an adaptation of the switched information cost of [8]. We call it Switched Multi-party Information Cost (SMIC).

► **Definition 5.1.** For a  $k$ -player protocol  $\pi$  with inputs drawn from  $\mu^n$  let

$$\text{SMIC}_{\mu^n}(\pi) = \sum_{i=1}^k (I(X_i; \Pi_i \mid MZ) + I(M; \Pi_i \mid X_i Z)) .$$

Note that the notion of SMIC is only defined with respect to the distribution  $\mu^n$  that we defined, and we may thus omit the distribution from the notation. We note that in order to simplify the expressions we often consider the public randomness as implicit in the information theoretic expressions we use below. It can be materialized either as part of the transcript or in the conditioning of the information theoretic expressions.

We can now prove the main result of this section.

► **Theorem 5.2.** For any fixed  $0 \leq \epsilon < \frac{1}{2}$ , for any protocol  $\pi$  externally  $\epsilon$ -computing  $\text{AND}_k$ ,

$$\text{SMIC}_{\mu}(\pi) = \Omega(k) .$$

## 6 The function Disjointness

In the  $k$  players  $n$ -bit disjointness function  $\text{Disj}_k^n$ , every player  $i \in \llbracket 1, k \rrbracket$  has an  $n$ -bit string  $(x_i^\ell)_{\ell \in \llbracket 1, n \rrbracket}$ , and the players have to output 1 if and only if there exists a coordinate  $\ell$  where all players have the bit 1. Formally,  $\text{Disj}_k^n(x) = \bigvee_{\ell=1}^n \bigwedge_{i=1}^k x_i^\ell$ .

### 6.1 Switched multi-party information cost of $\text{Disj}_k^n$

We first prove a direct-sum-type property which allows us to make the link between the functions  $\text{AND}_k$  and  $\text{Disj}_k^n$ . A similar property was proved in [8] in the coordinator model; our peer-to-peer model requires a different, more involved, construction, since we do not have the coordinator, and moreover no player can act as the coordinator since it would get too much information. Since  $\text{Disj}_k^n$  is the disjunction of  $n$   $\text{AND}_k$  functions, we analyze the switched multi-party information cost of  $\text{Disj}_k^n$  using the distribution  $\mu^n$ .

► **Lemma 6.1.** Let  $k > 3$ . For any protocol  $\pi$  externally  $\epsilon$ -computing  $\text{Disj}_k^n$ , there exists a protocol  $\pi'$  externally  $\epsilon$ -computing  $\text{AND}_k$  such that

$$\text{SMIC}_{\mu^n}(\pi) \geq n \cdot \text{SMIC}_{\mu}(\pi') .$$

Coupled with the lower bound on  $\text{SMIC}(\pi')$  for any protocol  $\pi'$  that computes  $\text{AND}_k$  (Section 5), the above lemma gives us a lower bound on  $\text{SMIC}(\pi)$  for any protocol that computes the function  $\text{Disj}_k^n$ :

► **Theorem 6.2.** *Let  $k > 3$ . Given any fixed  $0 \leq \epsilon < \frac{1}{2}$ , for any protocol  $\pi$  externally  $\epsilon$ -computing  $\text{Disj}_k^n$  it holds that*

$$\text{SMIC}_{\mu^n}(\pi) = \Omega(kn) .$$

## 6.2 Multi-party information complexity and communication complexity of $\text{Disj}_k^n$

The next lemma is key to our argument. The theorem that follows is a consequence of it and of Theorem 6.2.

► **Lemma 6.3.** *For any  $k$ -player protocol  $\pi$ ,  $\text{SMIC}_{\mu^n}(\pi) \leq \text{MIC}_{\mu^n}(\pi)$ .*

The next theorem follows immediately from Theorem 6.2 and Lemma 6.3.

► **Theorem 6.4.** *Let  $k > 3$ . Given any fixed  $0 \leq \epsilon < \frac{1}{2}$ , for any protocol  $\pi$  externally  $\epsilon$ -computing  $\text{Disj}_k^n$ , it holds that*

$$\text{MIC}_{\mu^n}(\pi) = \Omega(kn) .$$

We now conclude with a lower bound on the randomized communication complexity of the disjointness function.

► **Theorem 6.5.** *Given any fixed  $0 \leq \epsilon < \frac{1}{2}$ , there is a constant  $\alpha$  such that for  $n \geq \frac{1}{\alpha}k$ ,*

$$\text{CC}^\epsilon(\text{Disj}_k^n) = \Omega(kn) .$$

We note that our tight lower bound holds also for protocols where only one player is required to output the value of the function.

## 7 Randomness complexity of private protocols

A protocol  $\pi$  is said to *privately* compute a function  $f$  if, at the end of the execution of the protocol, the players have learned nothing but the value of that function. We now prove that the (information theoretic) private computation of  $\text{Disj}_k^n$  requires  $\Omega(n)$  random bits. We prove this result using the information theoretic results for  $\text{Disj}_k^n$  of the previous sections. The definitions and the details of the proof are deferred to the full version of the paper.

► **Theorem 7.1.** *Let  $k > 3$ . Then  $\mathcal{R}(\text{Disj}_k^n) = \Omega(n)$ , where  $\mathcal{R}(f)$  is the minimum number of random bits necessary for a protocol to privately compute  $f$ .*

## 8 Conclusions and open problems

We introduce new models and new information theoretic tools for the study of communication complexity, and other complexity measures, in the natural peer-to-peer, multi-party, number-in-hand setting. We prove a number of properties of our new models and measures, and exemplify their effectiveness by proving two lower bounds on communication complexity, as well as a lower bound on the amount of randomness necessary for certain private computations.

To the best of our knowledge, our lower bounds on communication complexity are the first tight (non-trivial) lower bounds on communication complexity in the natural *peer-to-peer* multi-party setting, and our lower bound on the randomness complexity of private computations is the first that grows with the size of the input, while the computed function is a boolean one (i.e., the size of the output does not grow with the size of the input).



We believe that our models and tools may find additional applications and may open the way to further study of the natural peer-to-peer setting and to the building of a more solid bridge between the the fields of communication complexity and of distributed computation.

Our work raises a number of questions. First, how can one relax the restrictions that we impose on the general asynchronous model and still prove communication complexity lower bounds in a peer-to-peer setting? Our work seems to suggest that novel techniques and ideas, possibly not based on information theory, are necessary for this task, and it would be most interesting to find those. Second, it would be interesting to identify the necessary and sufficient conditions that guarantee the “rectangularity” property of communication protocols in a peer-to-peer setting. While this property is fundamental to the analysis of two-party protocols, it turns out that once one turns to the multi-party peer-to-peer setting, not only does this property become subtle to prove, but also this property does not always hold. Given the central (and sometimes implicit) role of the rectangularity property in the literature, it would be interesting to identify when it holds in the multi-party peer-to-peer number-in-hand setting.

---

### References

- 1 Gilad Asharov and Yehuda Lindell. A Full Proof of the BGW Protocol for Perfectly Secure Multiparty Computation. *J. Cryptology*, 30(1):58–151, 2017. doi:10.1007/s00145-015-9214-4.
- 2 Reuven Bar-Yehuda, Benny Chor, Eyal Kushilevitz, and Alon Orlitsky. Privacy, additional information and communication. *IEEE Transactions on Information Theory*, 39(6):1930–1943, 1993. doi:10.1109/18.265501.
- 3 Ziv Bar-Yossef, T. S. Jayram, Ravi Kumar, and D. Sivakumar. An information statistics approach to data stream and communication complexity. *J. Comput. Syst. Sci.*, 68(4):702–732, 2004. doi:10.1016/j.jcss.2003.11.006.
- 4 Boaz Barak, Mark Braverman, Xi Chen, and Anup Rao. How to compress interactive communication. In *Proceedings of the 42nd ACM symposium on Theory of computing*, STOC '10, pages 67–76, New York, NY, USA, 2010. ACM. doi:10.1145/1806689.1806701.
- 5 Michael Ben-Or, Shafi Goldwasser, and Avi Wigderson. Completeness theorems for non-cryptographic fault-tolerant distributed computation. In *Proceedings of the twentieth annual ACM symposium on Theory of computing*, STOC '88, pages 1–10, New York, NY, USA, 1988. ACM. doi:10.1145/62212.62213.
- 6 C. Blundo, A. De Santis, G. Persiano, and U. Vaccaro. Randomness complexity of private computation. *computational complexity*, 8(2):145–168, 1999. doi:10.1007/s000370050025.
- 7 Mark Braverman. Interactive Information Complexity. *SIAM J. Comput.*, 44(6):1698–1739, 2015. doi:10.1137/130938517.
- 8 Mark Braverman, Faith Ellen, Rotem Oshman, Toniann Pitassi, and Vinod Vaikuntanathan. A Tight Bound for Set Disjointness in the Message-Passing Model. In *54th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2013, 26-29 October, 2013, Berkeley, CA, USA*, pages 668–677. IEEE Computer Society, 2013. doi:10.1109/FOCS.2013.77.
- 9 Mark Braverman, Ankit Garg, Denis Pankratov, and Omri Weinstein. From information to exact communication. In *Proceedings of the 45th annual ACM symposium on Symposium on theory of computing*, STOC '13, pages 151–160, New York, NY, USA, 2013. ACM. doi:10.1145/2488608.2488628.
- 10 Mark Braverman and Rotem Oshman. On Information Complexity in the Broadcast Model. In Chryssis Georgiou and Paul G. Spirakis, editors, *Proceedings of the 2015 ACM Symposium on Principles of Distributed Computing, PODC 2015, Donostia-San Sebastián, Spain, July 21 - 23, 2015*, pages 355–364. ACM, 2015. doi:10.1145/2767386.2767425.

- 11 Mark Braverman and Anup Rao. Information Equals Amortized Communication. *IEEE Trans. Information Theory*, 60(10):6058–6069, 2014. doi:10.1109/TIT.2014.2347282.
- 12 Amit Chakrabarti and Sagar Kale. Strong Fooling Sets for Multi-player Communication with Applications to Deterministic Estimation of Stream Statistics. In Irit Dinur, editor, *IEEE 57th Annual Symposium on Foundations of Computer Science, FOCS 2016, 9-11 October 2016, New Brunswick, New Jersey, USA*, pages 41–50. IEEE Computer Society, 2016. doi:10.1109/FOCS.2016.14.
- 13 Amit Chakrabarti, Subhash Khot, and Xiaodong Sun. Near-optimal lower bounds on the multi-party communication complexity of set disjointness. In *In IEEE Conference on Computational Complexity*, pages 107–117, 2003.
- 14 Amit Chakrabarti, Yaoyun Shi, Anthony Wirth, and Andrew Chi-Chih Yao. Informational Complexity and the Direct Sum Problem for Simultaneous Message Complexity. In *FOCS*, pages 270–278, 2001. doi:10.1109/SFCS.2001.959901.
- 15 Arkadev Chattopadhyay and Sagnik Mukhopadhyay. Tribes Is Hard in the Message Passing Model. In Ernst W. Mayr and Nicolas Ollinger, editors, *32nd International Symposium on Theoretical Aspects of Computer Science, STACS 2015, March 4-7, 2015, Garching, Germany*, volume 30 of *LIPICs*, pages 224–237. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2015. doi:10.4230/LIPICs.STACS.2015.224.
- 16 Arkadev Chattopadhyay and Toniann Pitassi. The Story of Set Disjointness. *SIGACT News*, 41(3):59–85, September 2010. doi:10.1145/1855118.1855133.
- 17 Arkadev Chattopadhyay, Jaikumar Radhakrishnan, and Atri Rudra. Topology Matters in Communication. In *55th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2014, Philadelphia, PA, USA, October 18-21, 2014*, pages 631–640, 2014. doi:10.1109/FOCS.2014.73.
- 18 Arkadev Chattopadhyay and Atri Rudra. The Range of Topological Effects on Communication. In Magnús M. Halldórsson, Kazuo Iwama, Naoki Kobayashi, and Bettina Speckmann, editors, *Automata, Languages, and Programming - 42nd International Colloquium, ICALP 2015, Kyoto, Japan, July 6-10, 2015, Proceedings, Part II*, volume 9135 of *Lecture Notes in Computer Science*, pages 540–551. Springer, 2015. doi:10.1007/978-3-662-47666-6\_43.
- 19 David Chaum, Claude Crépeau, and Ivan Damgard. Multiparty unconditionally secure protocols. In *Proceedings of the twentieth annual ACM symposium on Theory of computing, STOC '88*, pages 11–19, New York, NY, USA, 1988. ACM. doi:10.1145/62212.62214.
- 20 Danny Dolev and Tomás Feder. Multiparty Communication Complexity. In *30th Annual Symposium on Foundations of Computer Science, Research Triangle Park, North Carolina, USA, 30 October - 1 November 1989*, pages 428–433. IEEE Computer Society, 1989. doi:10.1109/SFCS.1989.63514.
- 21 Faith Ellen, Rotem Oshman, Toniann Pitassi, and Vinod Vaikuntanathan. Brief Announcement: Private Channel Models in Multi-party Communication Complexity. In *27th International Symposium on Distributed Computing (DISC), Jerusalem, Israel*, pages 575–576, 2013.
- 22 Uri Feige, Joe Killian, and Moni Naor. A Minimal Model for Secure Computation (Extended Abstract). In *Proceedings of the Twenty-sixth Annual ACM Symposium on Theory of Computing, STOC '94*, pages 554–563, New York, NY, USA, 1994. ACM. doi:10.1145/195058.195408.
- 23 Silvio Frischknecht, Stephan Holzer, and Roger Wattenhofer. Networks cannot compute their diameter in sublinear time. In Yuval Rabani, editor, *Proceedings of the Twenty-Third Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2012, Kyoto, Japan, January 17-19, 2012*, pages 1150–1162. SIAM, 2012. doi:10.1137/1.9781611973099.

- 24 Anna Gál and Parikshit Gopalan. Lower Bounds on Streaming Algorithms for Approximating the Length of the Longest Increasing Subsequence. *SIAM J. Comput.*, 39(8):3463–3479, 2010. doi:10.1137/090770801.
- 25 Anna Gál and Adi Rosén.  $\Omega(\log n)$  Lower Bounds on the Amount of Randomness in 2-Private Computation. *SIAM J. Comput.*, 34(4):946–959, 2005. doi:10.1137/S0097539703432785.
- 26 Andre Gronemeier. Asymptotically Optimal Lower Bounds on the NIH-Multi-Party Information Complexity of the AND-Function and Disjointness. In Susanne Albers and Jean-Yves Marion, editors, *26th International Symposium on Theoretical Aspects of Computer Science, STACS 2009, February 26-28, 2009, Freiburg, Germany, Proceedings*, volume 3 of *LIPICs*, pages 505–516. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, Germany, 2009. doi:10.4230/LIPICs.STACS.2009.1846.
- 27 Zengfeng Huang, Bozidar Radunovic, Milan Vojnovic, and Qin Zhang. Communication Complexity of Approximate Matching in Distributed Graphs. In Ernst W. Mayr and Nicolas Ollinger, editors, *32nd International Symposium on Theoretical Aspects of Computer Science, STACS 2015, March 4-7, 2015, Garching, Germany*, volume 30 of *LIPICs*, pages 460–473. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2015. doi:10.4230/LIPICs.STACS.2015.460.
- 28 T. S. Jayram. Hellinger Strikes Back: A Note on the Multi-party Information Complexity of AND. In *Proceedings of the 12th International Workshop and 13th International Workshop on Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, APPROX '09 / RANDOM '09*, pages 562–573, Berlin, Heidelberg, 2009. Springer-Verlag. doi:10.1007/978-3-642-03685-9\_42.
- 29 Bala Kalyanasundaram and Georg Schintger. The Probabilistic Communication Complexity of Set Intersection. *SIAM J. Discret. Math.*, 5(4):545–557, November 1992. doi:10.1137/0405044.
- 30 Iordanis Kerenidis, Adi Rosén, and Florent Urrutia. Multi-Party Protocols, Information Complexity and Privacy. In Piotr Faliszewski, Anca Muscholl, and Rolf Niedermeier, editors, *41st International Symposium on Mathematical Foundations of Computer Science, MFCS 2016, August 22-26, 2016 - Kraków, Poland*, volume 58 of *LIPICs*, pages 57:1–57:16. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2016. doi:10.4230/LIPICs.MFCS.2016.57.
- 31 Janne H. Korhonen and Jukka Suomela. Brief Announcement: Towards a Complexity Theory for the Congested Clique. In Andréa W. Richa, editor, *31st International Symposium on Distributed Computing, DISC 2017, October 16-20, 2017, Vienna, Austria*, volume 91 of *LIPICs*, pages 55:1–55:3. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2017. doi:10.4230/LIPICs.DISC.2017.55.
- 32 Janne H. Korhonen and Jukka Suomela. Towards a complexity theory for the congested clique. *CoRR*, abs/1705.03284, 2017. arXiv:1705.03284.
- 33 Eyal Kushilevitz and Yishay Mansour. Randomness in Private Computations. *SIAM J. Discrete Math.*, 10(4):647–661, 1997. doi:10.1137/S0895480196306130.
- 34 Eyal Kushilevitz and Noam Nisan. *Communication complexity*. Cambridge University Press, 1997.
- 35 Eyal Kushilevitz, Rafail Ostrovsky, and Adi Rosén. Characterizing Linear Size Circuits in Terms of Pricacy. *J. Comput. Syst. Sci.*, 58(1):129–136, 1999. doi:10.1006/jcss.1997.1544.
- 36 Peter Bro Miltersen, Noam Nisan, Shmuel Safra, and Avi Wigderson. On Data Structures and Asymmetric Communication Complexity. *J. Comput. Syst. Sci.*, 57(1):37–49, 1998. doi:10.1006/jcss.1998.1577.

- 37 Jeff M. Phillips, Elad Verbin, and Qin Zhang. Lower Bounds for Number-in-Hand Multiparty Communication Complexity, Made Easy. *SIAM J. Comput.*, 45(1):174–196, 2016. doi:10.1137/15M1007525.
- 38 A. A. Razborov. On the Distributional Complexity of Disjointness. *Theor. Comput. Sci.*, 106(2):385–390, December 1992. doi:10.1016/0304-3975(92)90260-M.
- 39 Atish Das Sarma, Stephan Holzer, Liah Kor, Amos Korman, Danupon Nanongkai, Gopal Pandurangan, David Peleg, and Roger Wattenhofer. Distributed Verification and Hardness of Distributed Approximation. *CoRR*, abs/1011.3049, 2010. arXiv:1011.3049.
- 40 C. E. Shannon. A mathematical theory of communication. *Bell system technical journal*, 27, 1948.
- 41 David P. Woodruff and Qin Zhang. Tight Bounds for Distributed Functional Monitoring. *CoRR*, abs/1112.5153, 2011. arXiv:1112.5153.
- 42 David P. Woodruff and Qin Zhang. When Distributed Computation does not Help. *CoRR*, abs/1304.4636, 2013. arXiv:1304.4636.
- 43 David P. Woodruff and Qin Zhang. An Optimal Lower Bound for Distinct Elements in the Message Passing Model. In *Proceedings of the Twenty-fifth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '14, pages 718–733, Philadelphia, PA, USA, 2014. Society for Industrial and Applied Mathematics. URL: <http://dl.acm.org/citation.cfm?id=2634074.2634128>.
- 44 Andrew Chi-Chih Yao. Protocols for Secure Computations (Extended Abstract). In *23rd Annual Symposium on Foundations of Computer Science, Chicago, Illinois, USA, 3-5 November 1982*, pages 160–164. IEEE Computer Society, 1982. doi:10.1109/SFCS.1982.38.



# A Schur Complement Cheeger Inequality

Aaron Schild<sup>1</sup>

University of California, Berkeley, CA, USA  
aschild@berkeley.edu

---

## Abstract

Cheeger’s inequality shows that any undirected graph  $G$  with minimum normalized Laplacian eigenvalue  $\lambda_G$  has a cut with conductance at most  $O(\sqrt{\lambda_G})$ . Qualitatively, Cheeger’s inequality says that if the mixing time of a graph is high, there is a cut that certifies this. However, this relationship is not tight, as some graphs (like cycles) do not have cuts with conductance  $o(\sqrt{\lambda_G})$ .

To better approximate the mixing time of a graph, we consider a more general object. Specifically, instead of bounding the mixing time with cuts, we bound it with cuts in graphs obtained by Schur complementing out vertices from the graph  $G$ . Combinatorially, these Schur complements describe random walks in  $G$  restricted to a subset of its vertices. As a result, all Schur complement cuts have conductance at least  $\Omega(\lambda_G)$ . We show that unlike with cuts, this inequality is tight up to a constant factor. Specifically, there is a Schur complement cut with conductance at most  $O(\lambda_G)$ .

**2012 ACM Subject Classification** Mathematics of computing → Spectra of graphs

**Keywords and phrases** electrical networks, Cheeger’s inequality, mixing time, conductance, Schur complements

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2019.65

**Acknowledgements** I want to thank Satish Rao, Nikhil Srivastava, and Rasmus Kyng for helpful discussions.

## 1 Introduction

In this paper, we give a constant-factor approximation for the minimum ratio of electrical conductance to volume of any pair of sets  $S_1, S_2$ . When  $S_1$  is the complement of  $S_2$ , for example, this quantity is what is classically called the *conductance* of the set  $S_1$ , so the ratio that we consider is less than conductance of the graph  $G$ . We obtain this constant-factor approximation by showing that the minimum electrical conductance-to-volume ratio is approximated within a constant factor by  $\lambda_G$ . Thus, our quantity closes the classical  $\sqrt{\lambda_G}$  gap present between the upper and lower bounds in Cheeger’s inequality.

In particular, we prove the following partitioning result, which relates  $1/\lambda_G$  to effective resistances between sets in the graph  $G$ . Think of the weighted graph  $G$  as an electrical network, where each edge represents a conductor with electrical conductance equal to its weight. For two sets of vertices  $S_1$  and  $S_2$ , obtain a graph  $H$  by contracting all vertices in  $S_1$  to a single vertex  $s_1$  and all vertices in  $S_2$  to a single vertex  $s_2$ . Let  $\text{Reff}_G(S_1, S_2)$  denote the effective resistance between the vertices  $s_1$  and  $s_2$  in the graph  $H$ . Then, we show the following in Appendix A:

► **Theorem 1.** *In any weighted graph  $G$ , there are two sets of vertices  $S_1$  and  $S_2$  for which  $\text{Reff}_G(S_1, S_2) \geq 1/(25600\lambda_G \min(\text{vol}_G(S_1), \text{vol}_G(S_2)))$ . Furthermore, for any pair of sets  $S'_1, S'_2$ ,  $\text{Reff}_G(S'_1, S'_2) \leq 2/(\lambda_G \min(\text{vol}_G(S'_1), \text{vol}_G(S'_2)))$ .*

---

<sup>1</sup> Supported by NSF grant CCF-1816861.



## 1.1 Relationship to Cheeger's Inequality

For a set of vertices  $S$ , let  $\phi_S$  denote the total weight of edges leaving  $S$  divided by the total degree of the vertices in  $S$ . Throughout the literature, this quantity is often called the conductance of  $S$ . To avoid confusion with electrical conductance, we call this quantity the *fractional conductance* of  $S$ . Let  $\phi_G$  denote the minimum fractional conductance of any set  $S$  with at most half of the volume (total vertex degree). Cheeger's inequality for graphs [3, 2] is as follows:

► **Theorem 2** (Cheeger's Inequality). *For any weighted graph  $G$ ,  $\lambda_G/2 \leq \phi_G \leq \sqrt{2\lambda_G}$ .*

Cheeger's inequality was originally introduced in the context of manifolds [6]. It is a fundamental primitive in graph partitioning [20, 15] and for upper bounding the mixing time of Markov chains [19]. Motivated by spectral partitioning, much work has been done on higher-order generalizations of Cheeger's inequality [13, 14]. The myriad of applications for Cheeger's inequality and generalizations of it [4, 21], along with the  $\sqrt{\lambda_G}$  gap between the upper and lower bounds, have led to a long line of work that seeks to improve the quality of the partition found when the spectrum has certain properties (for example, bounded eigenvalue gap [11] or when the graph has special structure [10].)

Here, we get rid of the  $\sqrt{\lambda_G}$  gap by taking a different approach. Instead of assuming special combinatorial or spectral structure of the input graph to obtain a tighter relationship between conductance and  $\lambda_G$ , we introduce a more general object than graph cuts that enable a tighter approximation to  $\lambda_G$ . Instead of just considering cuts in the given graph  $G$ , we consider cuts obtained by picking two disjoint sets of vertices  $S_1$  and  $S_2$ , computing the Schur complement of  $G$  onto  $S_1 \cup S_2$ , and looking at the cut consisting of all edges between  $S_1$  and  $S_2$  in that Schur complement. Let  $\rho_G$  be the minimum conductance of any such cut (defined formally in Section 2). We show that the minimum conductance of any such cut is a constant factor approximation to  $\lambda_G$ :

► **Theorem 3.** *Let  $G$  be a weighted graph. Then*

$$\lambda_G/2 \leq \rho_G \leq 25600\lambda_G$$

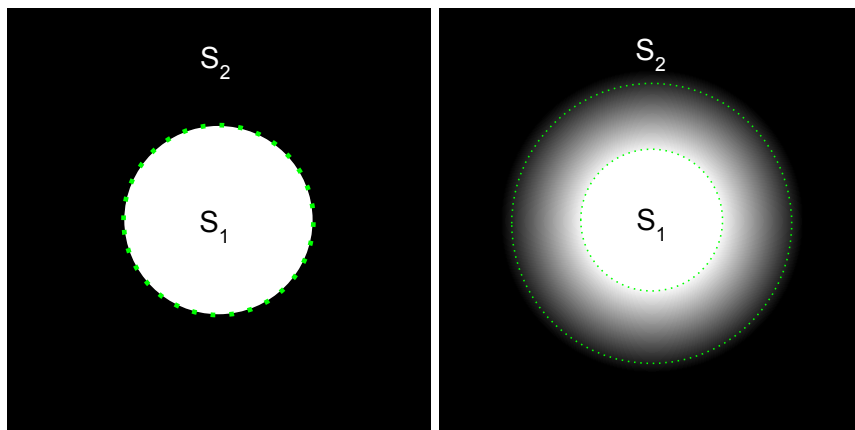
## 1.2 Graph Partitioning

Effective resistance in spectral graph theory has been used several times recently (for example [16, 1]) to obtain improved graph partitioning results.  $1/\lambda_G$  may not yield a good approximation to the effective resistance between pairs of vertices [5]. For example, on an  $n$ -vertex grid graph  $G$ , all effective resistances are between  $\Omega(1)$  and  $O(\log n)$ , but  $\lambda_G = \Theta(1/n)$ . Theorem 1 closes this gap by considering pairs of *sets* of vertices, not just pairs of vertices.

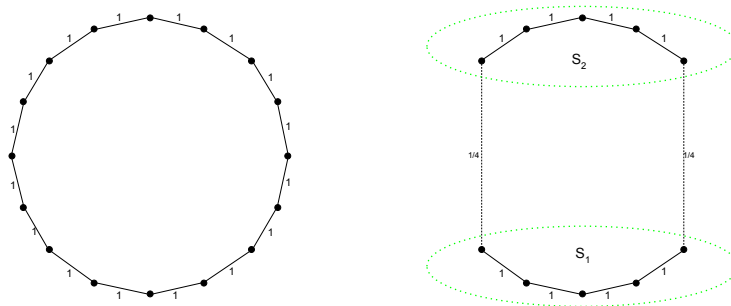
Cheeger's inequality is the starting point for analysis of spectral partitioning. In some partitioning tasks, cutting the graph does not make sense. For example, spectral partitioning is an important tool in image segmentation [18, 17]. Graph partitioning makes the most sense in image segmentation when one wants to find an object with a sharp boundary. However, in many images, like the one in Figure 1 on the right, objects may have fuzzy boundaries. In these cases, it is not clear which cut an image segmentation algorithm should return.

Considering cuts in Schur complements circumvents this ambiguity. Think of an image as a graph by making a vertex for each pixel and making an edge between adjacent pixels, where the weight on an edge is inversely related to the disparity between the colors of the endpoint pixels for the edge. An optimal segmentation in our setting would consist of the





■ **Figure 1** Spectral partitioning finds the  $S_1$ - $S_2$  cut in the left image, but may not in the right due to the presence of many equal weight cuts. The minimum fractional conductance Schur complement cut is displayed in both images.



■ **Figure 2** A tight example for the upper bound in Cheeger's inequality. The minimum fractional conductance of any cut in this graph is  $1/8$ , while the fractional conductance of the illustrated Schur complement cut on the right is  $2(1/4)/(2(1/4) + 8(1)) = 1/17 < 1/8$ .

two sets  $S_1$  and  $S_2$  corresponding to pixels on either side of the fuzzy boundary. Computing the Schur complement of the graph onto  $S_1 \cup S_2$  eliminates all vertices corresponding to pixels in the boundary.

Some examples in which Cheeger's inequality is not tight illustrate a similar phenomenon in which there are many equally good cuts. For example, let  $G$  be an unweighted  $n$ -vertex cycle. This is a tight example for the upper bound in Cheeger's inequality, as no cut has fractional conductance smaller than  $O(1/n)$  despite the fact that  $\lambda_G = \Theta(1/n^2)$ . Instead, divide the cycle into four equal-sized quarters and let  $S_1$  and  $S_2$  be two opposing quarters. The Schur complement cut between  $S_1$  and  $S_2$  has fractional conductance at most  $O(1/n^2)$ , which matches  $\lambda_G$  up to a constant factor.

### 1.3 Mixing Times of Markov Chains

Cheeger's inequality and variants of it can be used to upper bound the relaxation time of reversible Markov chains. The *relaxation time* of a reversible Markov chain is defined to be  $1/\lambda_G$  and approximates the mixing time of a chain up to a  $\log(1/\pi_{\min})$  factor, where  $\pi_{\min}$  is the minimum stationary distribution probability. To do this, one finds a fractional multicommodity flow that routes the demand  $\text{degree}(u)\text{degree}(v)$  for every pair of vertices

$u, v \in V(G)$ . The relevant quantity is the *congestion* of this flow. Specifically, view a multicommodity flow as a collection of paths  $\mathcal{P}$ , where a flow path has flow value  $f_p$ . The congestion of the flow is defined to be

$$\max_{e \in E(G)} \sum_{p \in \mathcal{P}: e \in p} |f_p|$$

The fractional conductance of any cut can be lower bounded using the the congestion of the flow. By the upper bound in Theorem 2, we get a lower bound on  $\lambda_G$  as well, which in turn yields an upper bound on the relaxation time of  $G$ . However, the congestion of the optimal fractional multicommodity flow is not a close approximation to the relaxation time of the graph, due to the loss in both Cheeger's inequality and in the gap between fractional multicommodity flows and sparsest cuts. One can improve upon this by considering a length-weighted form of congestion instead:

$$\max_{e \in E(G)} \sum_{p \in \mathcal{P}: e \in p} |f_p| |p|$$

The length-weighted congestion of any multicommodity flow in  $G$  routing the right demand is an upper bound on the relaxation time of  $G$ . For a precise statement of this, see Theorem 4.6 of [8]. This overcomes the square root present in the upper bound in Cheeger's inequality. However, this approach does not overcome the multicommodity flow-cut gap. In particular, any multicommodity flow in a constant-degree random Erdos-Renyi graph has length-weighted congestion at least  $\Omega(\log^2 n)$  (see, for example, Section 4.1 of [9]), despite the fact that these graphs have constant relaxation time.

Theorem 3 yields a upper lower bound on the relaxation time of a reversible Markov chain. Unlike the bounds discussed in the previous paragraph, it is tight up to a constant factor on all graphs. We do not know of a lower bound for Schur complement cut fractional conductance analogous to multicommodity flows for standard cuts.

## 2 Preliminaries

**Graph theory.** Consider an undirected, connected graph  $H$  with edge weights  $\{c_e^H\}_{e \in E(H)}$ ,  $m$  edges, and  $n$  vertices. Let  $V(H)$  and  $E(H)$  denote the vertex and edge sets of  $H$  respectively. For two sets of vertices  $A, B \subseteq V(H)$ , let  $E_H(A, B)$  denote the set of edges in  $H$  incident with one vertex in  $A$  and one vertex in  $B$  and let  $c^H(A, B) := \sum_{e \in E_H(A, B)} c_e^H$ . For a set of edges  $F \subseteq E(H)$ , let  $c^H(F) := \sum_{e \in F} c_e^H$ . For a set of vertices  $A \subseteq V(H)$ , let  $\partial_H A := E_H(A, V(H) \setminus A)$ . For a vertex  $v \in V(H)$ , let  $\partial_H v := \partial_H \{v\}$  denote the edges incident with  $v$  in  $H$  and let  $c_v^H := \sum_{e \in \partial_H v} c_e^H$ . For a set of vertices  $A \subseteq V(H)$ , let  $\text{vol}_H(A) := \sum_{v \in A} c_v^H$ . When  $A$  and  $B$  are disjoint, let  $H/(A, B)$  denote the graph with all vertices in  $A$  identified to one vertex  $a$  and all vertices in  $B$  identified to one vertex  $b$ . Formally, let  $H/(A, B)$  be the graph with  $V(H/(A, B)) = (V(H) \setminus (A \cup B)) \cup \{a, b\}$ , embedding  $f : V(H) \rightarrow V(H/(A, B))$  with  $f(u) := a$  if  $u \in A$ ,  $f(u) := b$  if  $u \in B$ , and  $f(u) := u$  otherwise, and edges  $\{f(u), f(v)\}$  for all  $\{u, v\} \in E(H)$ . Let  $H/A := H/(A, \emptyset)$ .

**Laplacians.** Let  $D_H$  be the  $n \times n$  diagonal matrix with rows and columns indexed by vertices in  $H$  and  $D_H(v, v) = c_v^H$  for all  $v \in V(H)$ . Let  $A_H$  be the adjacency matrix of  $H$ ; that is the matrix with  $A_H(u, v) = c_{uv}^H$  for all  $u, v \in V(H)$ . Let  $L_H := D_H - A_H$  be the Laplacian matrix of  $H$ . Let  $N_H := D_H^{-1/2} L_H D_H^{-1/2}$  denote the normalized Laplacian matrix

of  $H$ . For a matrix  $M$ , let  $M^\dagger$  denote the Moore-Penrose pseudoinverse of  $M$ . For subsets  $A$  and  $B$  of rows and columns of  $M$  respectively, let  $M[A, B]$  denote the  $|A| \times |B|$  submatrix of  $M$  restricted to those rows and columns. For a set of vertices  $S \subseteq V(H)$ , let  $\mathbf{1}_S$  denote the indicator vector for the set  $S$ . For two vertices  $u, v \in \mathbb{R}^n$ , let  $\chi_{uv} := \mathbf{1}_{\{u\}} - \mathbf{1}_{\{v\}}$ . When the graph is clear from context, we omit  $H$  from all of the subscripts and superscripts of  $H$ . For a vector  $x \in \mathbb{R}^n$ , let  $x_S \in \mathbb{R}^S$  denote the restriction of  $x$  to the coordinates in  $S$ .

Let  $\lambda_H$  denote the smallest nonzero eigenvalue of  $N_H$ . Equivalently,

$$\lambda_H := \min_{x \in \mathbb{R}^n: x^T D_H^{1/2} \mathbf{1}_{V(H)} = 0} \frac{x^T N_H x}{x^T x}$$

For any set of vertices  $X \subseteq V(H)$ , let

$$L_{\text{Schur}(H, X)} := L_H[X, X] - L_H[X, V(H) \setminus X] L_H[V(H) \setminus X, V(H) \setminus X]^{-1} L_H[V(H) \setminus X, X]$$

where brackets denote submatrices with the indexed rows and columns. The following fact applies specifically to Laplacian matrices:

► **Remark (Fact 2.3.6 of [12]).** For any graph  $H$  and any  $X \subseteq V(H)$ ,  $L_{\text{Schur}(H, X)}$  is the Laplacian matrix of an undirected graph.

Let  $\text{Schur}(H, X)$  denote the graph referred to in Remark 2. Schur complementation commutes with edge contraction and deletion and is associative:

► **Theorem 4** (Lemma 4.1 of [7], statement from [12]). *Given  $H$ ,  $S \subseteq V(H)$ , and any edge  $e$  with both endpoints in  $S$ ,*

$$\text{Schur}(H \setminus e, S) = \text{Schur}(H, S) \setminus e$$

and, for any pair of vertices  $x, y \in S$ ,

$$\text{Schur}(H/\{x, y\}, S) = \text{Schur}(H, S)/\{x, y\}$$

► **Theorem 5.** *Given  $H$  and two sets of vertices  $X \subseteq Y \subseteq V(H)$ ,  $\text{Schur}(\text{Schur}(H, Y), X) = \text{Schur}(H, X)$ .*

The following property follows from the definition of Schur complements:

► **Remark.** Let  $H$  be a graph and  $S \subseteq V(H)$ . Let  $I := \text{Schur}(H, S)$ . For any  $x \in \mathbb{R}^{V(H)}$  that is supported on  $S$  with  $x^T \mathbf{1}_{V(H)} = 0$ ,

$$x^T L_H^\dagger x = x_S^T L_I^\dagger x_S$$

The weight of edges in this graph can be computed using the following folklore fact, which we prove for completeness:

► **Theorem 6.** *For two disjoint sets  $C, D \subseteq V(H)$ , let  $I := \text{Schur}(H, C \cup D)$ . Then*

$$c^I(C, D) = \frac{1}{\chi_{cd}^T L_{H/(C, D)}^\dagger \chi_{cd}}$$

**Proof.** By definition,  $c^I(C, D) = c^{I/(C, D)}(\{c\}, \{d\})$ . By Theorem 4,  $I/(C, D) = \text{Schur}(H/(C, D), \{c, d\})$ . By Remark 2,  $c^{\text{Schur}(H/(C, D), \{c, d\})}(\{c\}, \{d\}) = \frac{1}{\chi_{cd}^T L_{H/(C, D)}^\dagger \chi_{cd}}$ . Combining these equalities gives the desired result. ◀

## 65:6 A Schur Complement Cheeger Inequality

We also use the following folklore fact about electrical flows, which we prove for the sake of completeness:

► **Theorem 7.** For two vertices  $s, t \in V(H)$ ,

$$\chi_{st}^T L_H^\dagger \chi_{st} = \frac{1}{\min_{p \in \mathbb{R}^{V(H)}: p_s \leq 0, p_t \geq 1} p^T L_H p}$$

**Proof.** We first show that

$$\chi_{st}^T L_H^\dagger \chi_{st} = \frac{1}{\min_{p \in \mathbb{R}^{V(H)}: p_s = 0, p_t = 1} p^T L_H p}$$

Taking the gradient of the objective  $p^T L_H p$  shows that that the optimal  $p$  are the potentials for an electrical flow with flow conservation at all vertices besides  $s$  and  $t$ . Therefore,  $p$  is proportional to  $L_H^\dagger \chi_{st} + \gamma \mathbf{1}$  for some  $\gamma \in \mathbb{R}$ . The constant of proportionality is  $\chi_{st}^T L_H^\dagger \chi_{st}$  since the  $s$ - $t$  potential drop in  $p$  is 1. Therefore,

$$\begin{aligned} \min_{p \in \mathbb{R}^{V(H)}: p_s = 0, p_t = 1} p^T L_H p &= \left( \frac{L_H^\dagger \chi_{st}}{\chi_{st}^T L_H^\dagger \chi_{st}} \right)^T L_H \left( \frac{L_H^\dagger \chi_{st}}{\chi_{st}^T L_H^\dagger \chi_{st}} \right) \\ &= \frac{1}{\chi_{st}^T L_H^\dagger \chi_{st}} \end{aligned}$$

The desired result follows from the fact that in the optimal  $p$ , all potentials are between 0 and 1 inclusive. ◀

**Notions of fractional conductance.** For a set of vertices  $A \subseteq V(H)$ , let

$$\phi_A^H := \frac{c^H(\partial_H(A))}{\min(\text{vol}_H(A), \text{vol}_H(V(H) \setminus A))}$$

be the *fractional conductance* of  $A$ . Let

$$\phi_H := \min_{A \subseteq V(H): A \neq \emptyset} \phi_A^H$$

be the *fractional conductance* of  $H$ .

For two disjoint sets of vertices  $A, B \subseteq V(H)$ , let  $I := \text{Schur}(H, A \cup B)$  and

$$\rho_{A,B}^H := \frac{c^I(A, B)}{\min(\text{vol}_I(A), \text{vol}_I(B))}$$

be the *Schur complement fractional conductance* of the pair of sets  $(A, B)$ . Define the *Schur complement fractional conductance* of the graph  $H$  to be

$$\rho_H := \min_{A, B \subseteq V(H): A \cap B = \emptyset, A \neq \emptyset, B \neq \emptyset} \rho_{A,B}^H$$

It will be helpful to deal with the quantities

$$\sigma_{A,B}^H := \frac{c^I(A, B)}{\min(\text{vol}_H(A), \text{vol}_H(B))}$$

and

$$\sigma_H := \min_{A, B \subseteq V(H): A \cap B = \emptyset, A \neq \emptyset, B \neq \emptyset} \sigma_{A,B}^H$$

as well, which we call the *mixed fractional conductances* of  $(A, B)$  and  $H$  respectively.

The following will be useful in relating  $\rho_{A,B}^H$  to  $\sigma_{A,B}^H$ :

► **Proposition 8.** *For any two sets  $X \subseteq Y \subseteq V(H)$ , let  $I := \text{Schur}(H, Y)$ . Then,*

$$\text{vol}_I(X) \leq \text{vol}_H(X)$$

**Proof.** It suffices to show this result when  $|X| = 1$  because  $\text{vol}$  is a sum of volumes (degrees) of vertices in the set. Furthermore, by Theorem 5, it suffices to show the result when  $|Y| = |V(H)| - 1$ . Let  $v$  be the unique vertex in  $H$  outside of  $Y$  and let  $u$  be the unique vertex in  $X$ . Then, by definition of the Schur complement,

$$\begin{aligned} \text{vol}_I(X) &= c_u^I \\ &= \sum_{w \in V(I)} c_{uw}^I \\ &= \sum_{w \in V(I)} \left( c_{uw}^H + \frac{c_{uv}^H c_{vw}^H}{c_v^H} \right) \\ &= \left( \sum_{w \in V(I)} c_{uw}^H \right) + \frac{c_{uv}^H}{c_v^H} \left( \sum_{w \in V(I)} c_{vw}^H \right) \\ &\leq \left( \sum_{w \in V(I)} c_{uw}^H \right) + c_{uv}^H \\ &= c_u^H \\ &= \text{vol}_H(X) \end{aligned}$$

as desired. ◀

To prove the upper bound, we given an algorithm for constructing a low fractional conductance Schur complement cut. The following result is helpful for making this algorithm take near-linear time:

► **Theorem 9** (Theorem 8.2 of [23]). *Given a graph  $H$ , there is a  $\tilde{O}(m)$ -time algorithm that produces a vector  $x \leftarrow \text{ApxFiedler}(H) \in \mathbb{R}^{V(H)}$  with  $x^T D_H^{1/2} \mathbf{1}_{V(H)} = 0$  for which*

$$x^T N_H x \leq 2\lambda_H x^T x$$

### 3 Lower bound

We now show the first inequality in Theorem 3, which follows from the following lemma by Proposition 8, which implies that  $\sigma_G \leq \rho_G$ .

► **Lemma 10.**

$$\lambda_G \leq 2\sigma_G$$

**Proof.** We lower bound the Schur complement fractional conductance of any pair of disjoint sets  $A, B \subseteq V(G)$ . Let  $I := \text{Schur}(G, A \cup B)$ . Let  $P$  be the  $(A \cup B) \times (A \cup B)$  diagonal matrix with  $P(u, u) = c_u^G$  for each  $u \in A \cup B$ . We start by lower bounding the minimum nonzero eigenvalue  $\lambda$  of the matrix  $P^{-1/2} L_I P^{-1/2}$ . Let  $\lambda_{\max}(M)$  denote the maximum eigenvalue of a symmetric matrix  $M$ . By definition of the Moore-Penrose pseudoinverse,

$$1/\lambda = \lambda_{\max}(P^{1/2} L_I^\dagger P^{1/2})$$

## 65:8 A Schur Complement Cheeger Inequality

By Remark 2,

$$\lambda_{\max}(P^{1/2}L_I^\dagger P^{1/2}) \leq \lambda_{\max}(N_G^\dagger) = 1/\lambda_G$$

Therefore,  $\lambda \geq \lambda_G$ . We now plug in a test vector. Let

$$z := P^{1/2} \left( \frac{\mathbf{1}_A}{\text{vol}_G(A)} - \frac{\mathbf{1}_B}{\text{vol}_G(B)} \right)$$

$z^T(P^{1/2}\mathbf{1}_{V(I)}) = 0$ , so

$$\begin{aligned} \lambda_G &\leq \lambda \\ &= \min_{x \in \mathbb{R}^{A \cup B}: x^T P^{1/2} \mathbf{1}_{V(I)} = 0} \frac{x^T (P^{-1/2} L_I P^{-1/2}) x}{x^T x} \\ &\leq \frac{z^T (P^{-1/2} L_I P^{-1/2}) z}{z^T z} \\ &= \frac{c^I(A, B) ((1/\text{vol}_G(A)) + (1/\text{vol}_G(B)))^2}{(\text{vol}_G(A)/\text{vol}_G(A)^2) + (\text{vol}_G(B)/\text{vol}_G(B)^2)} \\ &= \frac{c^I(A, B) \text{vol}_G(A \cup B)}{\text{vol}_G(A) \text{vol}_G(B)} \\ &\leq 2\sigma_{A, B}^G \end{aligned}$$

### 4 Upper bound

We now show the second inequality in Theorem 3:

► **Lemma 11.**

$$\rho_G \leq 25600\lambda_G$$

To prove this lemma, we need to find a pair of sets  $A$  and  $B$  with low Schur complement fractional conductance:

► **Lemma 12.** *There is a near-linear time algorithm  $\text{SweepCut}(G)$  that takes in a graph  $G$  with  $\lambda_G \leq 1/25600$  and outputs a pair of nonempty sets  $A$  and  $B$  with the following properties:*

- (Low Schur complement fractional conductance)  $\sigma_{A, B}^G \leq 640\lambda_G$
- (Large interior)  $\phi_A^G \leq 1/4$  and  $\phi_B^G \leq 1/4$

We now prove Lemma 11 given Lemma 12:

**Proof of Lemma 11 given Lemma 12.** Let  $I := \text{Schur}(G, A \cup B)$ . For any two vertices  $u, v \in A \cup B$ ,  $c_{uv}^I \geq c_{uv}^G$ . Therefore,  $\text{vol}_I(A) \geq 2 \sum_{u, v \in A} c_{uv}^G$  and  $\text{vol}_I(B) \geq 2 \sum_{u, v \in B} c_{uv}^G$ . By the “Large interior” guarantee of Lemma 12,  $2 \sum_{u, v \in A} c_{uv}^G \geq (3/4)\text{vol}_G(A)$  and  $2 \sum_{u, v \in B} c_{uv}^G \geq (3/4)\text{vol}_G(B)$ . Therefore,

$$\rho_{A, B}^G \leq 4/3\sigma_{A, B}^G \leq 1280\lambda_G$$

by the “Low Schur complement weight” guarantee when  $\lambda_G \leq 1/25600$ , as desired. When  $\lambda_G > 1/25600$ , the lemma is trivially true, as desired. ◀

Now, we implement **SweepCut**. The standard Cheeger sweep examines all thresholds  $q \in \mathbb{R}$  and for each threshold, computes the fractional conductance of the cut  $\partial S_{\leq q}$  of edges from vertices with eigenvector coordinate at most  $q$  to ones greater than  $q$ . Instead, the algorithm **SweepCut** examines all thresholds  $q \in \mathbb{R}$  and computes an upper bound (a proxy) for the  $\sigma_{S_{\leq q/2}, S_{\geq q}}^G$  for each positive  $q$  and  $\sigma_{S_{\leq q}, S_{\geq q/2}}^G$  for each negative  $q$ . Let  $I_q := \text{Schur}(G, S_{\geq q} \cup S_{\leq q/2})$  for  $q > 0$  and  $I_q := \text{Schur}(G, S_{\leq q} \cup S_{\geq q/2})$ . Let  $\kappa_q(y) := \min(q, \max(q/2, y))$  for  $q > 0$  and  $\kappa_q(y) = \min(q/2, \max(q, y))$  for  $q \leq 0$ . The proxy is the following quantity, which is defined for a specific shift of the Rayleigh quotient minimizer  $y \in \mathbb{R}^{V(G)}$ .

$$\widehat{c}^{I_q}(S_{\geq q}, S_{\leq q/2}) := \frac{4}{q^2} \sum_{e=uv \in E(G)} c_e^G (\kappa_q(y_u) - \kappa_q(y_v))^2$$

for  $q > 0$  and

$$\widehat{c}^{I_q}(S_{\leq q}, S_{\geq q/2}) := \frac{4}{q^2} \sum_{e=uv \in E(G)} c_e^G (\kappa_q(y_u) - \kappa_q(y_v))^2$$

for  $q \leq 0$ . We now show that this is indeed an upper bound:

► **Proposition 13.** *For all  $q > 0$ ,*

$$c^{I_q}(S_{\leq q/2}, S_{\geq q}) \leq \widehat{c}^{I_q}(S_{\leq q/2}, S_{\geq q})$$

For all  $q \leq 0$ ,

$$c^{I_q}(S_{\leq q}, S_{\geq q/2}) \leq \widehat{c}^{I_q}(S_{\leq q}, S_{\geq q/2})$$

**Proof.** We focus on the  $q > 0$ , as the reasoning for the  $q \leq 0$  case is the same. By Theorems 6 and 7,

$$c^{I_q}(S_{\leq q/2}, S_{\geq q}) = \min_{p \in \mathbb{R}^{V(G)}: p_a \leq 0 \forall a \in S_{\leq q/2}, p_a \geq 1 \forall a \in S_{\geq q}} p^T L_G p$$

The vector  $p$  with  $p_a := \frac{2}{q} \kappa_q(y_a) - 1$  for all vertices  $a \in V(G)$  is a feasible solution to the above optimization problem with objective value  $\widehat{c}^{I_q}(S_{\leq q/2}, S_{\geq q})$ . This is the desired result. ◀

This proxy allows us to relate Schur complement conductances together across different thresholds  $q$  in a similar proof to the proof of the upper bound of Cheeger's inequality given in [22]. One complication in our case is that Schur complements for different values of  $q$  overlap in their eliminated vertices. Our choice of  $\leq q/2, \geq q$  plays a key role here (as opposed to  $\leq 0, \geq q$ , for example) in ensuring that the overlap is small. We now give the algorithm **SweepCut**:



**Algorithm 1:** SweepCut( $G$ ).

---

**Input:** A graph  $G$  with  $\lambda_G \leq 1/25600$   
**Output:** Two sets of vertices  $A$  and  $B$  satisfying the guarantees of Lemma 12

- 1  $z \leftarrow$  vector with  $z^T N_G z \leq 2\lambda_G z^T z$  and  $z^T (D_G^{1/2} \mathbf{1}_{V(G)}) = 0$
- 2  $x \leftarrow D_G^{-1/2} z$
- 3  $y \leftarrow x - \alpha \mathbf{1}_{V(G)}$  for a value  $\alpha$  such that  $\text{vol}_G(\{v : y_v \leq 0\}) \geq \text{vol}_G(V(G))/2$  and  $\text{vol}_G(\{v : y_v \geq 0\}) \geq \text{vol}_G(V(G))/2$
- 4 **foreach**  $q \in \mathbb{R}$  **do**
- 5      $S_{\geq q} \leftarrow$  vertices with  $y_v \geq q$
- 6      $S_{\leq q} \leftarrow$  vertices with  $y_v \leq q$
- 7 **end**
- 8 **foreach**  $q > 0$  **do**
- 9     **if** (1)  $\widehat{c}^{I_q}(S_{\leq q/2}, S_{\geq q}) \leq 640\lambda_G \min(\text{vol}_G(S_{\leq q/2}), \text{vol}_G(S_{\geq q}))$ , (2)  $c^G(\partial S_{\geq q/2}) \leq 1/4 \text{vol}_G(S_{\geq q})$ , and (3)  $\phi_{S_{\geq q}} \leq 1/4$  **then**
- 10         **return**  $(S_{\leq q/2}, S_{\geq q})$
- 11     **end**
- 12 **end**
- 13 **foreach**  $q \leq 0$  **do**
- 14     **if** (1)  $\widehat{c}^{I_q}(S_{\geq q/2}, S_{\leq q}) \leq 640\lambda_G \min(\text{vol}_G(S_{\geq q/2}), \text{vol}_G(S_{\leq q}))$ , (2)  $c^G(\partial S_{\geq q/2}) \leq 1/4 \text{vol}_G(S_{\leq q})$ , and (3)  $\phi_{S_{\leq q}} \leq 1/4$  **then**
- 15         **return**  $(S_{\leq q}, S_{\geq q/2})$
- 16     **end**
- 17 **end**

---

Our analysis relies on the following key technical result, which we prove in Appendix B:

► **Proposition 14.** For any  $a, b \in \mathbb{R}$ ,

$$\int_0^\infty \frac{(\kappa_q(a) - \kappa_q(b))^2}{q} dq \leq 10(a - b)^2$$

**Proof of Lemma 12.**

**Algorithm well-definedness.** We start by showing that SweepCut returns a pair of sets.

Assume, for the sake of contradiction, that SweepCut does not return a pair of sets. Let  $I_q := \text{Schur}(G, S_{\geq q} \cup S_{\leq q/2})$  for  $q > 0$  and  $I_q := \text{Schur}(G, S_{\leq q} \cup S_{\geq q/2})$  for  $q \leq 0$ . By the contradiction assumption, for all  $q > 0$ ,

$$\text{vol}_G(S_{\geq q}) \leq \frac{\widehat{c}^{I_q}(S_{\geq q}, S_{\leq q/2})}{640\lambda_G} + 4c^G(\partial S_{\geq q}) + 4c^G(\partial S_{\leq q/2})$$

and for all  $q < 0$ ,

$$\text{vol}_G(S_{\leq q}) \leq \frac{\widehat{c}^{I_q}(S_{\leq q}, S_{\geq q/2})}{640\lambda_G} + 4c^G(\partial S_{\leq q}) + 4c^G(\partial S_{\geq q/2})$$

Since  $\sum_{v \in V(G)} c_v^G x_v = 0$ ,

$$\sum_{v \in V(G)} c_v^G x_v^2 \leq \sum_{v \in V(G)} c_v^G y_v^2$$

Now, we bound the positive  $y_v$  and negative  $y_v$  parts of this sum separately. Negating  $y$  shows that it suffices to bound the positive part. Order the vertices in  $S_{\geq 0}$  in decreasing order by  $y_v$  value. Let  $v_i$  be the  $i$ th vertex in this ordering, let  $k := |S_{\geq 0}|$ ,  $y_{k+1} := 0$ ,  $y_i := y_{v_i}$ ,  $c_i := c_{v_i}^G$ , and  $S_i := \{v_1, v_2, \dots, v_i\}$  for each integer  $i \in [k]$ . Then

$$\begin{aligned} \sum_{v \in S_{\geq 0}} c_v^G y_v^2 &= \sum_{i=1}^k c_i y_i^2 \\ &= \sum_{i=1}^k (\text{vol}_G(S_i) - \text{vol}_G(S_{i-1})) y_i^2 \\ &= \sum_{i=1}^k \text{vol}_G(S_i) (y_i^2 - y_{i+1}^2) \\ &= 2 \int_0^\infty \text{vol}_G(S_{\geq q}) q dq \end{aligned}$$

By our volume upper bound from up above,

$$\begin{aligned} 2 \int_0^\infty \text{vol}_G(S_{\geq q}) q dq &\leq 2 \int_0^\infty \frac{\widehat{c}^I q(S_{\geq q}, S_{\leq q/2})}{640 \lambda_G} q dq + 8 \int_0^\infty c^G(\partial S_{\geq q}) q dq + 8 \int_0^\infty c^G(\partial S_{\leq q/2}) q dq \\ &= 2 \int_0^\infty \frac{\widehat{c}^I q(S_{\geq q}, S_{\leq q/2})}{640 \lambda_G} q dq + 8 \int_0^\infty c^G(\partial S_{\geq q}) q dq + 8 \int_0^\infty c^G(\partial S_{> q/2}) q dq \\ &= 2 \int_0^\infty \frac{\widehat{c}^I q(S_{\geq q}, S_{\leq q/2})}{640 \lambda_G} q dq + 40 \int_0^\infty c^G(\partial S_{\geq q}) q dq \end{aligned}$$

Substitution and Proposition 14 show that

$$\begin{aligned} 2 \int_0^\infty \text{vol}_G(S_{\geq q}) q dq &\leq 8 \sum_{e=uv \in E(G)} c_e^G \int_0^\infty \left( \frac{(\kappa_q(y_u) - \kappa_q(y_v))^2}{640 \lambda_G q} + 5 \mathbb{1}_{q \in [y_u, y_v]} q \right) dq \\ &\leq 8 \sum_{e=uv \in E(G)} c_e^G \left( \frac{10}{640 \lambda_G} (y_u - y_v)^2 + 5 |y_u^2 - y_v^2| \right) \end{aligned}$$

By Cauchy-Schwarz,

$$\begin{aligned} 8 \sum_{e=uv \in E(G)} c_e^G \left( \frac{10}{640 \lambda_G} (y_u - y_v)^2 + 5 |y_u^2 - y_v^2| \right) &\leq \frac{1}{8 \lambda_G} \sum_{e=uv \in E(G)} c_e^G (y_u - y_v)^2 \\ &\quad + 40 \sqrt{\sum_{e=uv \in E(G)} c_e^G (y_u - y_v)^2} \sqrt{\sum_{e=uv \in E(G)} c_e^G (y_u + y_v)^2} \\ &\leq \frac{1}{4} \sum_{v \in V(G)} c_v^G x_v^2 \\ &\quad + 80 \sqrt{\lambda_G} \sqrt{\sum_{v \in V(G)} c_v^G x_v^2} \sqrt{\sum_{v \in V(G)} c_v^G y_v^2} \end{aligned}$$

But since  $\sum_{v \in V(G)} c_v^G x_v^2 \leq \sum_{v \in V(G)} c_v^G y_v^2$  and  $\lambda_G < 1/25600$ ,

$$\frac{1}{4} \sum_{v \in V(G)} c_v^G x_v^2 + 80\sqrt{\lambda_G} \sqrt{\sum_{v \in V(G)} c_v^G x_v^2} \sqrt{\sum_{v \in V(G)} c_v^G y_v^2} < \frac{1}{2} \sum_{v \in V(G)} c_v^G y_v^2.$$

Negating  $y$  shows that  $\sum_{v \in S_{\leq 0}} c_v^G y_v^2 < 1/2 \sum_{v \in V(G)} c_v^G y_v^2$  as well. But these statements cannot both hold; a contradiction. Therefore, **SweepCut** must output a pair of sets.

**Runtime.** Computing  $z$  takes  $\tilde{O}(m)$  time by Theorem 9. Therefore, it suffices to show that the foreach loops can each be implemented in  $O(m)$  time. This implementation is similar to the  $O(m)$ -time implementation of the Cheeger sweep.

We focus on the first foreach loop, as the second is the same with  $q$  negated. First, note that the functions  $\phi_{S_{\geq q}}$ ,  $c^G(\partial S_{\geq q/2})$ , and  $\text{vol}_G(S_{\geq q})$  of  $q$  are piecewise constant, with breakpoints at  $q = y_u$  and  $q = 2y_u$  for each  $u \in V(G)$ . Furthermore, these functions can be computed for all values in  $O(m)$  time using an  $O(m)$ -time Cheeger sweep for each function.

Therefore, it suffices to compute the value of  $\tilde{c}^{Iq}(S_{\leq q/2}, S_{\geq q})$  for all  $q \geq 0$  that are local minima in  $O(m)$  time. Let  $h(q) := \tilde{c}^{Iq}(S_{\leq q/2}, S_{\geq q})$ . Notice that the functions  $h(q)$  and  $h'(q)$  are piecewise quadratic and linear functions of  $q$  respectively, with breakpoints at  $q = y_u$  and  $q = 2y_u$ . Using five  $O(m)$ -time Cheeger sweeps, one can compute the  $q^2$ ,  $q$  and 1 coefficients of  $h(q)$  and the  $q$  and 1 coefficients of  $h'(q)$  between all pairs of consecutive breakpoints. After computing these coefficients, one can compute the value of each function at a point  $q$  in  $O(1)$  time. Furthermore, given two consecutive breakpoints  $a$  and  $b$ , one can find all points  $q \in (a, b)$  with  $h'(q) = 0$  in  $O(1)$  time. Each local minimum for  $h$  is either a breakpoint or a point with  $h'(q) = 0$ . Since  $h$  and  $h'$  have  $O(n)$  breakpoints, all local minima can be computed in  $O(n)$  time.  $h$  can be evaluated at all of these points in  $O(n)$  time. Therefore, all local minima of  $h$  can be computed in  $O(m)$  time. Since the algorithm does return a  $q$ , some local minimum for  $h$  also suffices, so this implementation produces the desired result in  $O(m)$  time.

**Low Schur complement fractional conductance.** By Proposition 13,

$$c^{Iq}(S_{\geq q}, S_{\leq q/2}) \leq \tilde{c}^{Iq}(S_{\geq q}, S_{\leq q/2})$$

Therefore,  $c^{Iq}(S_{\geq q}, S_{\leq q/2}) \leq 640\lambda_G \min(\text{vol}_G(S_{\geq q}), \text{vol}_G(S_{\leq q/2}))$  for  $q \geq 0$  by the foreach loop if condition. Repeating this reasoning for  $q < 0$  yields the desired result.

**Large interior.** By definition of  $\alpha$ ,  $\text{vol}_G(S_{\geq q}) \leq \text{vol}_G(S_{\leq q/2})$  for  $q > 0$ . Since  $c^G(\partial S_{\leq q/2}) \leq 1/4\text{vol}_G(S_{\geq q})$ ,  $\phi_{S_{\leq q/2}} \leq 1/4$ , as desired.  $\blacktriangleleft$

---

## References

- 1 Vedat Levi Alev, Nima Anari, Lap Chi Lau, and Shayan Oveis Gharan. Graph Clustering using Effective Resistance. In *9th Innovations in Theoretical Computer Science Conference, ITCS 2018, January 11-14, 2018, Cambridge, MA, USA*, pages 41:1–41:16, 2018. doi: 10.4230/LIPIcs.ITCS.2018.41.
- 2 N. Alon and V. D. Milman.  $\lambda_1$ , Isoperimetric Inequalities for Graphs, and Superconcentrators, 1985.
- 3 Noga Alon. Eigenvalues and expanders. *Combinatorica*, 6(2):83–96, 1986.

- 4 Afonso S. Bandeira, Amit Singer, and Daniel A. Spielman. A Cheeger Inequality for the Graph Connection Laplacian. *SIAM J. Matrix Analysis Applications*, 34(4):1611–1630, 2013.
- 5 Ashok K. Chandra, Prabhakar Raghavan, Walter L. Ruzzo, Roman Smolensky, and Prasoos Tiwari. The Electrical Resistance of a Graph Captures its Commute and Cover Times. *Computational Complexity*, 6(4):312–340, 1997.
- 6 Jeff Cheeger. A lower bound for the smallest eigenvalue of the Laplacian. In *Proceedings of the Princeton conference in honor of Professor S. Bochner*, pages 195–199, 1969.
- 7 Charles J. Colbourn, Robert P. J. Day, and Louis D. Nel. Unranking and Ranking Spanning Trees of a Graph. *J. Algorithms*, 10(2):271–286, 1989.
- 8 Venkatesan Guruswami. Rapidly Mixing Markov Chains: A Comparison of Techniques (A Survey). *CoRR*, abs/1603.01512, 2016.
- 9 Nabil Kahale. A semidefinite bound for mixing rates of Markov chains. In William H. Cunningham, S. Thomas McCormick, and Maurice Queyranne, editors, *Integer Programming and Combinatorial Optimization*, pages 190–203, Berlin, Heidelberg, 1996. Springer Berlin Heidelberg.
- 10 Jonathan A. Kelner, James R. Lee, Gregory N. Price, and Shang-Hua Teng. Metric uniformization and spectral bounds for graphs. *CoRR*, abs/1008.3594, 2010.
- 11 Tsz Chiu Kwok, Lap Chi Lau, Yin Tat Lee, Shayan Oveis Gharan, and Luca Trevisan. Improved Cheeger’s Inequality: Analysis of Spectral Partitioning Algorithms Through Higher Order Spectral Gap. In *Proceedings of the Forty-fifth Annual ACM Symposium on Theory of Computing*, STOC ’13, pages 11–20, New York, NY, USA, 2013. ACM. doi:10.1145/2488608.2488611.
- 12 Rasmus Kyng. *Approximate Gaussian Elimination*. PhD thesis, Yale University, 2017.
- 13 James R. Lee, Shayan Oveis Gharan, and Luca Trevisan. Multi-way Spectral Partitioning and Higher-order Cheeger Inequalities. In *Proceedings of the Forty-fourth Annual ACM Symposium on Theory of Computing*, STOC ’12, pages 1117–1130, New York, NY, USA, 2012. ACM. doi:10.1145/2213977.2214078.
- 14 Anand Louis, Prasad Raghavendra, Prasad Tetali, and Santosh Vempala. Many Sparse Cuts via Higher Eigenvalues. In *Proceedings of the Forty-fourth Annual ACM Symposium on Theory of Computing*, STOC ’12, pages 1131–1140, New York, NY, USA, 2012. ACM. doi:10.1145/2213977.2214079.
- 15 Ulrike Luxburg. A Tutorial on Spectral Clustering. *Statistics and Computing*, 17(4):395–416, December 2007. doi:10.1007/s11222-007-9033-z.
- 16 Aleksander Madry, 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, SODA 2015, San Diego, CA, USA, January 4-6, 2015*, pages 2019–2036, 2015. doi:10.1137/1.9781611973730.134.
- 17 Marina Meila and Jianbo Shi. Learning Segmentation by Random Walks. In *Advances in Neural Information Processing Systems 13, Papers from Neural Information Processing Systems (NIPS) 2000, Denver, CO, USA*, pages 873–879, 2000. URL: <http://papers.nips.cc/paper/1830-learning-segmentation-by-random-walks>.
- 18 Jianbo Shi and Jitendra Malik. Normalized Cuts and Image Segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 22(8):888–905, August 2000. doi:10.1109/34.868688.
- 19 Alistair Sinclair. Improved bounds for mixing rates of Markov chains and multicommodity flow. *Combinatorics, Probability and Computing*, 1:351–370, 1992.
- 20 Daniel A. Spielman and Shang-Hua Teng. Nearly Linear Time Algorithms for Preconditioning and Solving Symmetric, Diagonally Dominant Linear Systems. *SIAM J. Matrix Analysis Applications*, 35(3):835–885, 2014. doi:10.1137/090771430.

## 65:14 A Schur Complement Cheeger Inequality

- 21 John Steenbergen, Caroline Klivans, and Sayan Mukherjee. A Cheeger-type inequality on simplicial complexes. *Advances in Applied Mathematics*, 56:56–77, 2014. doi:10.1016/j.aam.2014.01.002.
- 22 Luca Trevisan. Lecture 4 from cs359g: Graph partitioning and expanders, Stanford University. <http://theory.stanford.edu/~trevisan/cs359g/lecture04.pdf>, January 2011.
- 23 Nisheeth K. Vishnoi.  $Lx = b$ . *Foundations and Trends in Theoretical Computer Science*, 8(1-2):1–141, 2013. doi:10.1561/04000000054.

### A Proof of Theorem 1

**Proof of Theorem 1.** For any two sets of vertices  $S_1, S_2$  in a graph  $G$ ,

$$\text{Reff}_G(S_1, S_2) \min(\text{vol}_G(S_1), \text{vol}_G(S_2)) = \frac{1}{\sigma_{S_1, S_2}^G}$$

Therefore, the desired result follows from Lemmas 11 and 10. ◀

### B Proof of Proposition 14

**Proof of Proposition 14.** Without loss of generality, suppose that  $a \leq b$ . We break the analysis up into cases:

**Case 1:  $a \leq 0$ .** In this case,  $\kappa_q(a) = q/2$  for all  $q \geq 0$ , so

$$\begin{aligned} \int_0^\infty \frac{(\kappa_q(a) - \kappa_q(b))^2}{q} dq &= \int_0^b \frac{(q/2 - q)^2}{q} dq \\ &\quad + \int_b^{2b} \frac{(q/2 - b)^2}{q} dq \\ &\quad + \int_{2b}^\infty \frac{(q/2 - q/2)^2}{q} dq \\ &= \frac{b^2}{8} + \int_b^{2b} (q/4 - b + b^2/q) dq \\ &= \frac{b^2}{2} - b^2 + b^2(\ln 2) \\ &\leq 10(a - b)^2 \end{aligned}$$

as desired.

Case 2:  $a > 0$  and  $b \leq 2a$ . In this case,

$$\begin{aligned}
 \int_0^\infty \frac{(\kappa_q(a) - \kappa_q(b))^2}{q} dq &= \int_0^a \frac{(q-q)^2}{q} dq \\
 &+ \int_a^b \frac{(a-q)^2}{q} dq \\
 &+ \int_b^{2a} \frac{(a-b)^2}{q} dq \\
 &+ \int_{2a}^{2b} \frac{(q/2-b)^2}{q} dq \\
 &+ \int_{2b}^\infty \frac{(q/2-q/2)^2}{q} dq \\
 &\leq \int_a^{2b} \frac{(a-b)^2}{q} dq \\
 &= (a-b)^2 \ln(2b/a) \\
 &\leq (a-b)^2 \ln 4 \leq 10(a-b)^2
 \end{aligned}$$

as desired.

Case 3:  $a > 0$  and  $b > 2a$ . In this case,

$$\begin{aligned}
 \int_0^\infty \frac{(\kappa_q(a) - \kappa_q(b))^2}{q} dq &= \int_0^a \frac{(q-q)^2}{q} dq \\
 &+ \int_a^{2a} \frac{(a-q)^2}{q} dq \\
 &+ \int_{2a}^b \frac{(q/2-q)^2}{q} dq \\
 &+ \int_b^{2b} \frac{(q/2-b)^2}{q} dq \\
 &+ \int_{2b}^\infty \frac{(q/2-q/2)^2}{q} dq \\
 &\leq \int_a^{2b} \frac{(q/2-q)^2}{q} dq \\
 &\leq b^2/2 \\
 &\leq 2(a-b)^2 \leq 10(a-b)^2
 \end{aligned}$$

as desired. ◀






# Game Efficiency Through Linear Programming Duality

Nguyễn Kim Thăng

IBISC, Univ Evry, University Paris Saclay, Evry, France

thang@ibisc.fr

 <https://orcid.org/0000-0002-6085-9453>

---

## Abstract

The efficiency of a game is typically quantified by the price of anarchy (PoA), defined as the worst ratio of the value of an equilibrium – solution of the game – and that of an optimal outcome. Given the tremendous impact of tools from mathematical programming in the design of algorithms and the similarity of the price of anarchy and different measures such as the approximation and competitive ratios, it is intriguing to develop a duality-based method to characterize the efficiency of games.

In the paper, we present an approach based on linear programming duality to study the efficiency of games. We show that the approach provides a general recipe to analyze the efficiency of games and also to derive concepts leading to improvements. The approach is particularly appropriate to bound the PoA. Specifically, in our approach the dual programs naturally lead to competitive PoA bounds that are (almost) optimal for several classes of games. The approach indeed captures the smoothness framework and also some current non-smooth techniques/concepts. We show the applicability to the wide variety of games and environments, from congestion games to Bayesian welfare, from full-information settings to incomplete-information ones.

**2012 ACM Subject Classification** Theory of computation → Algorithmic game theory and mechanism design

**Keywords and phrases** Price of Anarchy, Primal-Dual

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2019.66

**Related Version** A full version of the paper is available at [32], <https://arxiv.org/abs/1708.06499>.

**Funding** Research supported by the ANR project OATA n° ANR-15-CE40-0015-01.

**Acknowledgements** We thank Tim Roughgarden for helpful feedback and comments.

## 1 Introduction

Algorithmic Game Theory – a domain at the intersection of Game Theory and Algorithms – has been extensively studied in the last two decades. The development of the domain, as well as those of many other research fields, have witnessed a common phenomenon: interesting notions, results have been flourished at the early stage, then deep methods, techniques have been established at a more mature stage leading to further achievements. In Algorithmic Game Theory, a representative illustration is the notion and results on the price of anarchy and the smoothness argument method [24]. In a game, the price of anarchy (PoA) [15] is defined as the worst ratio between the cost of a Nash equilibrium and that of an optimal solution. The PoA is now considered as standard and is the most popular



© Nguyễn Kim Thăng;

licensed under Creative Commons License CC-BY

10th Innovations in Theoretical Computer Science (ITCS 2019).

Editor: Avrim Blum; Article No. 66; pp. 66:1–66:20

Leibniz International Proceedings in Informatics



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

measure to characterize the inefficiency of Nash equilibria – solutions of a game – in the same sense of approximation ratio in Approximation Algorithms and competitive ratio in Online Algorithms.

Mathematical programming in general and linear programming in particular are powerful tools in many research fields. Among others, linear programming has a tremendous impact on the design of algorithms. Linear programming and duality play crucial and fundamental roles in several elegant methods such as primal-dual and dual-fitting in Approximation Algorithms [34] and online primal-dual framework [6] in Online Algorithms. Given the similarity of the notions of PoA, approximation and competitive ratios, it is intriguing and also desirable to develop a method based on duality to characterize the PoA of games. In this paper, we present and aim at developing a framework based on linear programming duality to study the efficiency of games.

### 1.1 A primal-dual approach

In high-level, the approach follows the standard primal-dual or dual-fitting techniques in approximation/online algorithms. The approach consists of associating a game to an underlying optimization problem and formulate an integer program corresponding to the optimization problem. Next consider the linear program by relaxing the integer constraints and its dual LP. Note that until this step, no notion of game has been intervened. Then given a Nash equilibrium, construct dual variables in such a way that one can relate the dual objective to the cost of the Nash equilibrium. The PoA is then bounded by the ratio between the primal objective (essentially, the cost of the Nash equilibrium) and the dual objective (a lower bound of the optimum cost by weak duality). This approach has been considered by Kulkarni and Mirrokni [17] for full-information games with convex objectives.

There are two crucial steps in the approach. First, by this method, the bound of PoA is at least as large as the integrality gap of the formulation. Hence, to prove optimal PoA one has to derive a formulation (of the corresponding optimization problem) whose the integrality gap matches to the optimal PoA. This is very similar to the issue of linear-programming-based approaches in Approximation/Online Algorithms. Note that this issue is a main obstacle in [17] in order to study non-convex objectives (see discussion in Section 1.3). The second crucial step is the construction of dual variables. The dual variables need to reflect the notion of Nash equilibria as well as their properties in order to relate to the costs of equilibria. Intuitively, to prove optimal bound on the PoA, the constructed dual variables must constitute an optimal dual solution.

To overcome these obstacles, in the paper we systematically consider *configuration* linear programs and a primal-dual approach. Given a problem (game), we first consider a natural formulation of the problem. Then, the approach consists of introducing exponential variables and constraints to the natural formulation to get a configuration LP. The additional constraints have intuitive and simple interpretations: one constraint guarantees that the game admits exactly one outcome and the other constraint ensures that if a player uses a strategy then this strategy must be a component of the outcome. As the result, the configuration LPs significantly improve the integrality gap over that of the natural formulations.

The configuration LPs have been considered in approximation algorithms and to the best of our knowledge, the main approach is rounding. Here, to study the efficiency of games, we consider a primal-dual approach. The primal-dual approach is very appropriate to study the PoA through the mean of configuration LPs. In the dual of the configuration programs, the dual constraints naturally lead to the construction of dual variables and the

PoA bounds. Intuitively, one dual constraint corresponds exactly to the definition of Nash equilibrium and the other dual constraint settles the PoA bounds. Note that our approach gives stronger formulations and leads to more general results than that in [17] (see Section 1.3 for a discussion in more details).

## 1.2 Overview of Results

We illustrate the potential and the wide applicability of the approach throughout various results in the contexts of complete and incomplete-information environments, from the settings of congestion games to welfare maximization. The approach allows us to unify several previous results and establish new ones beyond the current techniques. It is worthy to note that the analyses are simple and are guided by dual LP very much in the sense of primal-dual methods in designing algorithms. Moreover, under the lens of LP duality, the notion of smooth games in both full-information settings [24] and incomplete-information settings [25, 31], the recent notion of no-envy learning [10] and the new notion of dual smooth (in this paper) can be naturally derived, which lead to the optimal bounds of the PoA of several games.

### 1.2.1 Smooth Games in Full-Information Settings

We first revisit smooth games by the primal-dual approach and show that the primal-dual approach captures the smoothness framework [24]. Roughgarden [24] has introduced the smoothness framework, which became quickly a standard technique, and showed that every  $(\lambda, \mu)$ -smooth game has a PoA of at most  $\lambda/(1 - \mu)$ . Through the duality approach, we show that in terms of techniques to study the PoA for complete information settings, the LP duality and the smoothness framework are exactly the same thing. Specifically, one of the dual constraint corresponds exactly to the definition of smooth games given in [24].

► **Informal Theorem 1.** *The primal-dual approach captures the smoothness framework in full-information settings.*

### 1.2.2 Congestion Games

We consider fundamental classes of *congestion games* in which we revisit and unify results in the atomic, non-atomic congestion games and prove the optimal PoA bound of coarse correlated equilibria in splittable congestion games.

**Atomic congestion games.** In this class, although the PoA bound follows the results for smooth games (Informal Theorem 1), we provide another configuration formulation and a similar primal-dual approach. The purpose of this formulation is twofold. First it shows the flexibility of the primal-dual approach. Second, it sets up the ground for an unified approach to other classes of congestion games.

**Non-atomic congestion games.** In this class, we re-prove the *optimal* PoA bound [29]. Along the line toward the optimal PoA bound for non-atomic congestion games, the equilibrium characterization by a variational inequality is at the core of the analyses [29, 9, 8]. In our proof, we establish the optimal PoA directly by the means of LP duality. By the LP duality as the unified approach, one can clearly observe that the non-atomic setting is a version of the atomic setting in large games (in the sense of [12]) in which each player weight becomes negligible (hence, the PoA of the atomic congestion games tend to that of

non-atomic ones). Besides, an advantage with LP approaches is that one can benefit from powerful techniques that have been developing for linear programming. Concretely, using the general framework on resource augmentation and primal-dual recently presented [19], we manage to recover and extend a resource augmentation result related to non-atomic setting [28].

► **Informal Theorem 2.** *In every non-atomic congestion game, for any constant  $r > 0$ , the cost of an equilibrium is at most  $1/r$  the optimum of the underlying optimization problem in which each demand is multiplied by a factor  $(1 + r)$ .*

**Splittable congestion games.** Roughgarden and Schoppmann [26] has presented a *local* smoothness property, a refinement of the smoothness framework, and proved that every  $(\lambda, \mu)$ -local-smooth splittable game has a PoA of  $\lambda/(1 - \mu)$ . This bound is tight for a large class of scalable cost functions in splittable games and holds for PoA of pure, mixed, correlated equilibria. However, this bound does not hold for coarse correlated equilibria and it remains an intriguing open question raised in [26]. Building upon the resilient ideas of non-atomic and atomic settings, we define a notion, called *dual smoothness*, which is inspired by the dual constraints. This new notion indeed leads to the *tight* PoA bound for coarse correlated equilibria in splittable games for a large class of cost functions; that answers the question in [26]. Note that the matching lower bound is given in [26] and that holds even for pure equilibria.

► **Definition 3.** A cost function  $\ell : \mathbb{R}^+ \rightarrow \mathbb{R}^+$  is  $(\lambda, \mu)$ -dual-smooth if for every vectors  $\mathbf{u} = (u_1, \dots, u_n)$  and  $\mathbf{v} = (v_1, \dots, v_n)$ ,

$$v\ell(u) + \sum_{i=1}^n u_i(v_i - u_i) \cdot \ell'(u) \leq \lambda \cdot v\ell(v) + \mu \cdot u\ell(u)$$

where  $u = \sum_{i=1}^n u_i$  and  $v = \sum_{i=1}^n v_i$ . A splittable congestion game is  $(\lambda, \mu)$ -dual-smooth if for every resource  $e$  in the game, function  $\ell_e$  is  $(\lambda, \mu)$ -dual-smooth.

► **Informal Theorem 4.** *The price of anarchy of coarse correlated equilibria of a splittable congestion game  $G$  is at most  $\inf_{(\lambda, \mu)} \lambda/(1 - \mu)$  where the infimum is taken over  $(\lambda, \mu)$  such that  $G$  is  $(\lambda, \mu)$ -dual-smooth. This bound is tight for the class of scalable cost functions.*

### 1.2.3 Welfare Maximization

We next consider the inefficiency of Bayes-Nash equilibria in the context of welfare maximization in incomplete-information environments.

**Smooth Auctions.** The notion of smooth auctions in incomplete-information settings, inspired by the original smoothness framework [24], has been introduced by Roughgarden [25], Syrgkanis and Tardos [31]. This powerful notion has been widely used to study the PoA of Bayes-Nash equilibria (see the recent survey [27]). We show that the primal-dual approach captures the smoothness framework in incomplete-information settings. In other words, the notion of smooth auctions can be naturally derived from dual constraints in the primal-dual approach.

► **Informal Theorem 5.** *The primal-dual approach captures the smoothness framework in incomplete-information settings.*

**Simultaneous Item-Bidding Auctions: Beyond Smoothness.** Many PoA bounds in auctions are settled by smoothness-based proofs. However, there are PoA bounds for auctions proved via non-smooth techniques and these techniques seem more powerful than the smoothness framework in such auctions. Representative examples are the simultaneous first- and second-price auctions where players' valuations are sub-additive. Feldman et al. [11] have proved that the PoA is constant while the smooth argument gives only logarithmic guarantees. We show that in this context, our approach is beyond the smoothness framework and also captures the non-smooth arguments in [11] by re-establishing their results. Specifically, a main step in our analysis – proving the feasibility of a dual constraint – corresponds exactly to a crucial claim in [11]. From this point of view, the primal-dual approach helps to identify the key steps in settling the PoA bounds.

► **Informal Theorem 6** ([11]). *Assume that players have independent distributions over sub-additive valuations. Then, every Bayes-Nash equilibrium of a first-price auction and of a second price auction has expected welfare at least  $1/2$  and  $1/4$  of the maximal welfare, respectively.*

Subsequently, we illuminate the potential of the primal-dual approach in formulating new concepts. Concretely, Daskalakis and Syrgkanis [10] have very recently introduced *no-envy learning dynamic* – a novel concept of learning in auctions. Note that when players have fractionally sub-additive (XOS) valuations<sup>1</sup>, no-envy outcomes are a relaxation of no-regret outcomes. No-envy dynamics have advantages over no-regret dynamics. In particular, no-envy outcomes maintain the approximate welfare optimality of no-regret outcomes while ensuring the computational tractability. Perhaps surprisingly, there is a connection between the primal-dual approach and no-envy dynamics. Indeed, the latter can be naturally derived from the dual constraints very much in the same way as the smoothness argument is. We show this connection by revisiting the following theorem by the means of the primal-dual approach.

► **Informal Theorem 7** ([10]). *Assume that players have XOS valuations. Then, every no-envy dynamic has the average welfare at least half the expected optimal welfare.*

**Sequential Auctions.** To illustrate the applicability of the primal-dual approach, we consider thereafter another format of auctions – sequential auctions. In a simple model of sequential auctions, items are sold one-by-one via single-item auctions. Sequential auctions has a long and rich literature [16] and sequentially selling items leads to complex issues in analyzing PoA. Leme et al. [18], Syrgkanis and Tardos [30] have studied sequential auctions for matching markets and matroid auctions in complete and incomplete-information settings in which at each step, an item is sold via the first-price auctions. In this paper, we consider the sequential auctions for sponsored search via the second-price auctions. Informally, auctioneer sells advertising slots one-by-one in the non-increasing order of click-through-rates (from the most attractive to the least one). At each step, players submit bid for the currently-selling slot and the highest-bid player receives the slot and pays the second highest bid. In the auction, we study the PoA of perfect Bayesian equilibria and show the following PoA bound for the sponsored search problem.

<sup>1</sup> A valuation  $v(\cdot)$  is XOS if there exists a family of vectors  $\mathcal{W} = (w^\ell)_\ell$  where  $w^\ell \in \mathbb{R}_+^m$  such that  $v(S) = \max_{w^\ell \in \mathcal{W}} \sum_{j \in S} w_j^\ell \forall S \subset [m]$ . The class XOS is a subset of sub-additive functions and is a superset of sub-modular functions.

► **Informal Theorem 8.** *The PoA of sequential second-price auctions for the sponsored search problem is at most 2.*

Note that among all auction formats for the sponsored search problem, the best known PoA guarantee [7] is 2.927 which has been achieved in generalized second price (GSP) auctions. An observation is that although the behaviour of players in sequential auctions might be complex, the performance guarantee is better than the currently best-known one in GSP auctions for the sponsored search problem. Consequently, this result shows that the efficiency of sequential auctions is not necessarily worse than the GSP ones and using primal-dual approach, analyzing sequential auctions is not necessarily harder than analyzing GSP ones neither.

Building upon the resilient ideas for the sponsored search problem, we provide an improved PoA bound of 2 for the matching market problem where the best known PoA bound is  $2e/(e-1) \approx 3.16$  due to Syrgkanis and Tardos [30]. That also answers a question raised in [30] whether the PoA in the incomplete-information settings must be strictly larger than the best-known PoA bound (which is 2) in the full-information settings.

► **Informal Theorem 9.** *The PoA of sequential first-price auctions for the matching market problem is at most 2.*

Due to the space limit, the results in sequential auctions can be found in the full paper available online [32].

### 1.3 Related works

As the main point of the paper is to emphasize the primal-dual approach to study game efficiency, in this section we mostly concentrate on currently existing methods. Results related to specific problems will be summarized in the corresponding sections.

The most closely related to our work is a recent result [17]. In their approach, Kulkarni and Mirrokni [17] considered a convex formulation of a given game and its dual program based on Fenchel duality. Then, given a Nash equilibrium, the dual variables are constructed by relating the cost of the Nash equilibrium to that of the dual objective. In high-level, our approach has the same idea as [17] and both approaches indeed have inspired by the standard primal-dual and dual-fitting in the design of algorithms. Our approach is distinguished to that in [17] in the two following aspects. First, we consider arbitrary (non-decreasing) objective functions and make use of configuration LPs in order to reduce substantially the integrality gap while the approach in [17] needs convex objective functions. In term of approaches based on mathematical programs in approximation algorithms, we have come up with stronger formulations than those in [17] – a crucial point toward optimal bounds. Second, we have shown a wide applicability of our approach from full-information environments to incomplete-information ones while the approach in [17] dealt only with full-information settings. A question has been raised in a the recent survey [27] is whether the framework in [17] could be extended to incomplete-information settings. Our primal-dual approach tends to answer that question.

The connection between LP duality and the PoA have been previously considered by Nadav and Roughgarden [22] and Bilo [5]. Both papers follow an approach which is different to ours. Roughly speaking, given a game they consider corresponding natural formulations and incorporate the equilibrium constraint directly to the primal (whereas in our approach the equilibrium constraint appears naturally in the dual). However, this approach encounters also the integrality-gap obstacle when one considers pure Nash equilibria and the objectives are non-linear or non-convex.

For the problems studied in the paper, we systematically strengthen natural LPs by the construction of configuration LPs presented in [20]. Makarychev and Sviridenko [20] propose a scheme that consists in solving the new LPs (with exponential number of variables) and rounding the fractional solutions to integer ones using decoupling inequalities for optimization problems. Instead of rounding techniques, we consider a primal-dual approach which is more adequate to studying game efficiency.

The smoothness framework has been introduced by Roughgarden [24]. This simple, elegant framework gives tight bounds for many classes of games in full-information settings including the celebrated atomic congestion games (and others in [24, 2]). Subsequently, Roughgarden and Schoppmann [26] presented a similar notion, called local-smoothness, to study the PoA of splittable games in which players can split their flow to arbitrarily small amounts and route the amounts in different manners. The local-smoothness is also powerful. It has been used to settle the PoA for a large class of cost functions in splittable games [26] and in opinion formation games [3].

The smoothness framework has been extended to incomplete-information environments by Roughgarden [25], Syrgkanis and Tardos [31]. It has successfully yielded tight PoA bounds for several widely-used auction formats. We recommend the reader to a very recent survey [27] for applications of the smoothness framework in incomplete-information settings. However, the smoothness argument has its limit. As mentioned earlier, the most illustrative examples are the simultaneous first and second price auctions where players' valuations are sub-additive. Feldman et al. [11] have proved that the PoA is constant while the smooth argument gives only logarithmic guarantees. An interesting open direction, as raised in [27], is to develop new approaches beyond the smoothness framework.

Linear programming (and mathematical programming in general) has been a powerful tool in the development of game theory. There is a vast literature on this subject. One of the most interesting recent treatments on the role of linear programming in game theory is the book [33]. Vohra [33] revisited fundamental results in mechanism design in an elegant manner by the means of linear programming and duality. It is surprising to see that many results have been shaped nicely by LPs.

## 2 Smooth Games under the Lens of Duality

In this section, we consider smooth games [24] in the point of view of configuration LPs and duality. In a game, each player  $i$  selects a strategy  $s_i$  from a set  $\mathcal{S}_i$  for  $1 \leq i \leq n$  and that forms a *strategy profile*  $\mathbf{s} = (s_1, \dots, s_n)$ . The cost  $C_i(\mathbf{s})$  of player  $i$  is a function of the strategy profile  $\mathbf{s}$  – the chosen strategies of all players. A *pure Nash equilibrium* is a strategy profile  $\mathbf{s}$  such that no player can decrease its cost via a unilateral deviation; that is, for every player  $i$  and every strategy  $s'_i \in \mathcal{S}_i$ ,  $C_i(\mathbf{s}) \leq C_i(s'_i, \mathbf{s}_{-i})$  where  $\mathbf{s}_{-i}$  denotes the strategies chosen by all players other than  $i$  in  $\mathbf{s}$ . The notion of Nash equilibrium is extended to the following more general equilibrium concepts.

A *mixed Nash equilibrium* [23] of a game is a product distribution  $\sigma = \sigma_1 \times \dots \times \sigma_n$  where  $\sigma_i$  is a probability distribution over the strategy set of player  $i$  such that no player can decrease its expected cost under  $\sigma$  via a unilateral deviation:  $\mathbb{E}_{\mathbf{s} \sim \sigma} [C_i(\mathbf{s})] \leq \mathbb{E}_{\mathbf{s}_{-i} \sim \sigma_{-i}} [C_i(s'_i, \mathbf{s}_{-i})]$  for every  $i$  and  $s'_i \in \mathcal{S}_i$ , where  $\sigma_{-i}$  is the product distribution of all  $\sigma_{i'}$ 's other than  $\sigma_i$ . A *correlated equilibrium* [1] of a game is a joint probability distribution  $\sigma$  over the strategy profile of the game such that  $\mathbb{E}_{\mathbf{s} \sim \sigma} [C_i(\mathbf{s}) | s_i] \leq \mathbb{E}_{\mathbf{s} \sim \sigma} [C_i(s'_i, \mathbf{s}_{-i}) | s_i]$  for every  $i$  and  $s_i, s'_i \in \mathcal{S}_i$ . Finally, a *coarse correlated equilibrium* [21] of a game is a joint probability distribution  $\sigma$



over the strategy profile of the game such that  $\mathbb{E}_{\mathbf{s} \sim \sigma}[C_i(\mathbf{s})] \leq \mathbb{E}_{\mathbf{s} \sim \sigma}[C_i(s'_i, \mathbf{s}_{-i})]$  for every  $i$  and  $s'_i \in \mathcal{S}_i$ . These notions of equilibria are presented in the order from the least to the most general ones and a notion captures the previous one as a strict subset.

The notion of smooth games and robust price of anarchy are given in [24]. A game with a joint cost objective function  $C(\mathbf{s}) = \sum_{i=1}^n C_i(\mathbf{s})$  is  $(\lambda, \mu)$ -smooth if for every two outcomes  $\mathbf{s}$  and  $\mathbf{s}^*$ ,

$$\sum_{i=1}^n C_i(s_i^*, \mathbf{s}_{-i}) \leq \lambda \cdot C(\mathbf{s}^*) + \mu \cdot C(\mathbf{s})$$

The *robust price of anarchy* of a game  $G$  is

$$\rho(G) := \inf \left\{ \frac{\lambda}{1 - \mu} : \text{the game is } (\lambda, \mu)\text{-smooth where } \mu < 1 \right\}$$

► **Theorem 10** ([24]). *For every game  $G$  with robust PoA  $\rho(G)$ , every coarse correlated equilibrium  $\sigma$  of  $G$  and every strategy profile  $\mathbf{s}^*$ ,*

$$\mathbb{E}_{\mathbf{s} \sim \sigma}[C(\mathbf{s})] \leq \rho(G) \cdot C(\mathbf{s}^*)$$

Until the end of the section, we revisit this theorem by our primal-dual approach.

**Formulation.** Given a game, we formulate the corresponding optimization problem by a configuration LP. Let  $x_{ij}$  be variable indicating whether player  $i$  chooses strategy  $s_{ij} \in \mathcal{S}_i$ . Informally, a *configuration*  $A$  in the formulation is a strategy profile of the game. Formally, a configuration  $A$  consists of pairs  $(i, j)$  such that  $(i, j) \in A$  means that in configuration  $A$ ,  $x_{ij} = 1$ . (In other words, in this configuration, player  $i$  selects strategy  $s_{ij} \in \mathcal{S}_i$ .) For every configuration  $A$ , let  $z_A$  be a variable such that  $z_A = 1$  if and only if  $x_{ij} = 1$  for all  $(i, j) \in A$ . Intuitively,  $z_A = 1$  if configuration  $A$  is the outcome of the game. For each configuration  $A$ , let  $c(A)$  be the cost of the outcome (strategy profile) corresponding to configuration  $A$ . Consider the following formulation and the dual of its relaxation.

$$\begin{array}{ll} \min \sum_A c(A)z_A & \max \sum_i \alpha_i + \beta \\ \sum_{j: s_{ij} \in \mathcal{S}_i} x_{ij} \geq 1 & \forall i \\ \sum_A z_A = 1 & \\ \sum_{A: (i,j) \in A} z_A = x_{ij} & \forall i, j \\ x_{ij}, z_A \in \{0, 1\} & \forall i, j, A \end{array} \quad \begin{array}{ll} & \alpha_i \leq \gamma_{ij} \quad \forall i, j \\ \beta + \sum_{(i,j) \in A} \gamma_{ij} \leq c(A) & \forall A \\ & \alpha_i \geq 0 \quad \forall i \end{array}$$

In the formulation, the first constraint ensures that a player  $i$  chooses a strategy  $s_{ij} \in \mathcal{S}_i$ . The second constraint means that there must be an outcome of the game. The third constraint guarantees that if a player  $i$  selects some strategy  $s_{ij}$  then the outcome configuration  $A$  must contain  $(i, j)$ .

**Construction of dual variables.** Assuming that the game is  $(\lambda, \mu)$ -smooth. Fix the parameters  $\lambda$  and  $\mu$ . Given a (arbitrary) coarse correlated equilibrium  $\sigma$ , define dual variables as follows:

$$\alpha_i := \frac{1}{\lambda} \mathbb{E}_{\mathbf{s} \sim \sigma} [C_i(\mathbf{s})], \quad \beta := -\frac{\mu}{\lambda} \mathbb{E}_{\mathbf{s} \sim \sigma} [C(\mathbf{s})], \quad \gamma_{ij} := \frac{1}{\lambda} \mathbb{E}_{\mathbf{s} \sim \sigma} [C_i(s_{ij}, \mathbf{s}_{-i})].$$

Informally, up to some constant factors depending on  $\lambda$  and  $\mu$ ,  $\alpha_i$  is the cost of player  $i$  in equilibrium  $\sigma$ ,  $-\beta$  stands for the cost of the game in equilibrium  $\sigma$  and  $\gamma_{ij}$  represents the cost of player  $i$  if player  $i$  uses strategy  $s_{ij}$  while other players  $i' \neq i$  follows strategies in  $\sigma$ . We notice that  $\beta$  has negative value.

**Feasibility.** We show that the constructed dual variables form a feasible solution. The first constraint follows exactly the definition of (coarse correlated) equilibrium. The second constraint is exactly the smoothness definition. Specifically, let  $\mathbf{s}^*$  be the strategy profile corresponding to configuration  $A$ . Note that  $\mathbb{E}_{\mathbf{s} \sim \sigma} [C_i(\mathbf{s}^*)] = C_i(\mathbf{s}^*)$ . The dual constraint reads

$$-\frac{\mu}{\lambda} \mathbb{E}_{\mathbf{s} \sim \sigma} [C(\mathbf{s})] + \sum_i \frac{1}{\lambda} \mathbb{E}_{\mathbf{s} \sim \sigma} [C_i(s_i^*, \mathbf{s}_{-i})] \leq \mathbb{E}_{\mathbf{s} \sim \sigma} [C_i(\mathbf{s}^*)]$$

which is the definition of  $(\lambda, \mu)$ -smoothness by arranging the terms and removing the expectation.

**Price of Anarchy.** By weak duality, the optimal cost among all outcomes of the problem (strategy profiles of the game) is at least the dual objective of the constructed dual variables. Hence, in order to bound the PoA, we will bound the ratio between the cost of an (arbitrary) equilibrium  $\sigma$  and the dual objective of the corresponding dual variables. The cost of equilibrium  $\sigma$  is  $\mathbb{E}_{\mathbf{s} \sim \sigma} [C(\mathbf{s})]$  while the dual objective of the constructed dual variables is

$$\sum_{i=1}^n \frac{1}{\lambda} \mathbb{E}_{\mathbf{s} \sim \sigma} [C_i(\mathbf{s})] - \frac{\mu}{\lambda} \mathbb{E}_{\mathbf{s} \sim \sigma} [C(\mathbf{s})] = \frac{1-\mu}{\lambda} \mathbb{E}_{\mathbf{s} \sim \sigma} [C(\mathbf{s})]$$

Therefore, for a  $(\lambda, \mu)$ -smooth game, the PoA is at most  $\lambda/(1-\mu)$ .

**Remark.** Having shown in [24], Theorem 10 applies also to outcome sequences generated by repeated play such as vanishing average regret. By the same duality approach, we can also recover this result (by setting dual variables related to the average cost during the play).

### 3 Splittable Congestion Games

**Model.** In this section we consider the splittable congestion games in discrete setting. Fix a constant  $\epsilon > 0$  (arbitrarily small). In a splittable congestion game, there is a set  $E$  of resources, each resource is associated to a non-decreasing differentiable cost function  $\ell_e : \mathbb{R}^+ \rightarrow \mathbb{R}^+$  such that  $x\ell_e(x)$  is convex. There are  $n$  players, a player  $i$  has a set of strategies  $\mathcal{S}_i$  and has weight  $w_i$ , a multiple of  $\epsilon$ . A strategy of player  $i$  is a distribution  $u^i$  of its weight  $w_i$  among strategies  $s_{ij}$  in  $\mathcal{S}_i$  such that  $\sum_{s_{ij} \in \mathcal{S}_i} u_{s_{ij}}^i = w_i$  and  $u_{s_{ij}}^i \geq 0$  is a multiple of  $\epsilon$ . A strategy profile is a vector  $\mathbf{u} = (u^1, \dots, u^n)$  of all players' strategies. We abuse notation and define  $u_e^i = \sum_{e \in s_{ij}} u_{s_{ij}}^i$  as the load player  $i$  distributes on resource  $e$  and

$u_e = \sum_{i=1}^n u_e^i$  the total load on  $e$ . Given a strategy profile  $\mathbf{u}$ , the cost of player  $i$  is defined as  $C_i(\mathbf{u}) := \sum_e u_e^i \cdot \ell_e(u_e)$ . A strategy profile  $\mathbf{u}$  is a pure Nash equilibrium if and only if for every player  $i$  and all  $s_{ij}, s_{ij'} \in \mathcal{S}_i$  with  $u_{s_{ij}}^i > 0$ :

$$\sum_{e \in s_{ij}} (\ell_e(u_e) + u_e^i \cdot \ell'_e(u_e)) \leq \sum_{e \in s_{ij'}} (\ell_e(u_e) + u_e^i \cdot \ell'_e(u_e))$$

The proof of this equilibrium characterization can be found in [13]. Again, the more general concepts of mixed, correlated and coarse correlated equilibria are defined similarly as in Section 2. In the game, the social cost is defined as  $C(\mathbf{u}) := \sum_{i=1}^n C_i(\mathbf{u}) = \sum_e u_e \ell_e(u_e)$ .

The PoA bounds has been recently established for a large class of cost functions by Roughgarden and Schoppmann [26]. The authors proposed a *local smoothness* framework and showed that the local smoothness arguments give optimal PoA bounds for a large class of cost functions in splittable congestion games. Prior to Roughgarden and Schoppmann [26], the works of Cominetti et al. [8] and Harks [13] have also the flavour of local smoothness though their bounds are not tight. The local smooth arguments extends to the correlated equilibria of a game but not to the coarse correlated equilibria. Motivating by the duality approach, we define a new notion of smoothness and prove a bound on the PoA of coarse correlated equilibria. It turns out that this PoA bound for coarse correlated equilibria is indeed *tight* for all classes of scale-invariant cost functions by the lower bound given by Roughgarden and Schoppmann [26, Section 5]. A class of cost function  $\mathcal{L}$  is *scale-invariant* if  $\ell \in \mathcal{L}$  implies that  $a \cdot \ell(b \cdot x) \in \mathcal{L}$  for every  $a, b > 0$ .

**Formulation.** Given a splittable congestion game, we formulate the problem by the same configuration program for non-atomic congestion game. Denote a finite set of multiples of  $\epsilon$  as  $\{a_0, a_1, \dots, a_m\}$  where  $a_k = k \cdot \epsilon$  and  $m = \max_{i=1}^n w_i / \epsilon$ . We say that  $T_e$  is a *configuration* of a resource  $e$  if  $T_e = \{(i, k) : 1 \leq i \leq n, 0 \leq k \leq m\}$  in which a couple  $(i, k)$  specifies the player ( $i$ ) and the amount  $a_k$  of the weight  $w_i$  that player  $i$  distributes to some strategy  $s_{ij} \in \mathcal{S}_i$  where  $e \in s_{ij}$ . Intuitively, a configuration of a resource is a strategy profile of a game restricted on the resource. Let  $x_{ijk}$  be variable indicating whether player  $i$  distributes an amount  $a_k$  of its weight to strategy  $s_{ij} \in \mathcal{S}_i$ . For every resource  $e$  and a configuration  $T_e$  on resource  $e$ , let  $z_{e, T_e}$  be a variable such that  $z_{e, T_e} = 1$  if and only if for  $(i, k) \in T_e$ ,  $x_{ijk} = 1$  for some  $s_{ij} \in \mathcal{S}_i$  such that  $e \in s_{ij}$ . For a configuration  $T_e$  of resource  $e$ , denote  $w(T_e)$  the total amount distributed by players in  $T_e$  to  $e$ .

$$\begin{array}{ll} \min \sum_{e, T_e} w(T_e) \ell_e(w(T_e)) z_{e, T_e} & \max \sum_i w_i \alpha_i + \sum_e \beta_e \\ \sum_{j, k} a_k x_{ijk} = w_i & \forall i \\ \sum_{T_e} z_{e, T_e} = 1 & \forall e \\ \sum_{T_e: (i, k) \in T_e} z_{e, T_e} = \sum_{j: e \in s_{ij}} x_{ijk} & \forall (i, k), e \\ x_{ij}, z_{e, T_e} \in \{0, 1\} & \forall i, j, e, T_e \end{array} \quad \begin{array}{l} a_k \alpha_i \leq \sum_{e: e \in s_{ij}} \gamma_{i, k, e} \\ \forall i, k, j \\ \beta_e + \sum_{(i, k) \in T_e} \gamma_{i, k, e} \leq w(T_e) \ell_e(w(T_e)) \\ \forall e, T_e \end{array}$$

Again, in the primal, the first constraint says that a player  $i$  distributes the total weight  $w_i$  among its strategies. The second constraint means that a resource  $e$  is always associated to a

configuration (possibly empty). The third constraint guarantees that if a player  $i$  distributes an amount  $a_k$  to some strategy  $s_{ij}$  containing resource  $e$  then there must be a configuration  $T_e$  such that  $(i, k) \in T_e$  and  $z_{e, T_e} = 1$ .

All previous duality proofs have the same structure: in the dual LP, the first constraint gives the characterization of an equilibrium and the second one settles the PoA bounds. Following this line, we give the following definition.

► **Definition 11.** A cost function  $\ell : \mathbb{R}^+ \rightarrow \mathbb{R}^+$  is  $(\lambda, \mu)$ -dual-smooth if for every vectors  $\mathbf{u} = (u_1, \dots, u_n)$  and  $\mathbf{v} = (v_1, \dots, v_n)$ ,

$$v\ell(u) + \sum_{i=1}^n u_i(v_i - u_i) \cdot \ell'(u) \leq \lambda \cdot v\ell(v) + \mu \cdot u\ell(u)$$

where  $u = \sum_i u_i$  and  $v = \sum_i v_i$ . A splittable congestion game is  $(\lambda, \mu)$ -dual-smooth if every resource  $e$  in the game, function  $\ell_e$  is  $(\lambda, \mu)$ -dual-smooth.

► **Theorem 12.** For every  $(\lambda, \mu)$ -dual-smooth splittable congestion game  $G$ , the price of anarchy of coarse correlated equilibria of  $G$  is at most  $\lambda/(1 - \mu)$ . This bound is tight for the class of scalable cost functions.

**Proof.** The proof follows the duality scheme.

**Dual Variables.** Fix parameter  $\lambda$  and  $\mu$ . Given a coarse correlated equilibrium  $\sigma$ , define corresponding dual variables as follows.

$$\begin{aligned} \alpha_i &= \frac{1}{\lambda} \mathbb{E}_{\mathbf{u} \sim \sigma} \left[ \sum_{e \in s_{ij}} \ell_e(u_e) + u_e^i \ell'_e(u_e) \right] \text{ for some } s_{ij} \in \mathcal{S}_i : u_{s_{ij}}^i > 0, \\ \beta_e &= -\frac{1}{\lambda} \mathbb{E}_{\mathbf{u} \sim \sigma} \left[ \mu \cdot u_e \ell_e(u_e) + \sum_i (u_e^i)^2 \cdot \ell'_e(u_e) \right], \\ \gamma_{i,k,e} &= \frac{1}{\lambda} \mathbb{E}_{\mathbf{u} \sim \sigma} [a_k (\ell_e(u_e) + u_e^i \ell'_e(u_e))]. \end{aligned}$$

The dual variables have similar interpretations as previous analysis. Up to some constant factors, variable  $\alpha_i$  is the marginal cost of a strategy used by player  $i$  in the equilibrium; and  $\gamma_{i,k,e}$  represents an estimation of the cost of player  $i$  on resource  $e$  if player  $i$  distributes an amount  $a_k$  of its weight to some strategy containing  $e$  while players  $i'$  other than  $i$  follows their strategies in the equilibrium.

**Feasibility.** By this definition of dual variables, the first dual constraint holds since it is the definition of coarse correlated equilibrium. Rearranging the terms, the second dual constraint for a resource  $e$  and a configuration  $T_e$  reads

$$\frac{1}{\lambda} \sum_{(i,k) \in T_e} \mathbb{E}_{\mathbf{u} \sim \sigma} [a_k \cdot \ell_e(u_e) + u_e^i (a_k - u_e^i) \ell'_e(u_e)] \leq w(T_e) \ell_e(w(T_e)) + \frac{\mu}{\lambda} \mathbb{E}_{\mathbf{u} \sim \sigma} [u_e \ell_e(u_e)]$$

This inequality follows directly from the definition of  $(\lambda, \mu)$ -dual-smoothness and linearity of expectation (and note that  $w(T_e) \ell_e(w(T_e)) = \mathbb{E}_{\mathbf{u} \sim \sigma} [w(T_e) \ell_e(w(T_e))]$  and  $w(T_e) = \sum_{(i,k) \in T_e} a_k$ ).

**Bounding primal and dual.** By the definition of dual variables, the dual objective is

$$\begin{aligned}
 \sum_i w_i \alpha_i + \sum_e \beta_e &= \sum_e \left( \sum_i u_e^i \alpha_i + \beta_e \right) \\
 &= \frac{1}{\lambda} \mathbb{E}_{\mathbf{u} \sim \sigma} \left[ \sum_e u_e \ell_e(u_e) + \sum_i (u_e^i)^2 \cdot \ell'_e(u_e) \right] - \frac{1}{\lambda} \mathbb{E}_{\mathbf{u} \sim \sigma} \left[ \mu \cdot \sum_e u_e \ell_e(u_e) + \sum_i (u_e^i)^2 \cdot \ell'_e(u_e) \right] \\
 &= \frac{1 - \mu}{\lambda} \mathbb{E}_{\mathbf{u} \sim \sigma} \left[ \sum_e u_e \ell_e(u_e) \right]
 \end{aligned}$$

while the cost of the equilibrium  $\sigma$  is  $\mathbb{E}_{\mathbf{u} \sim \sigma} [\sum_e u_e \ell_e(u_e)]$ . The theorem follows.  $\blacktriangleleft$

#### 4 Efficiency in Welfare Maximization

In a general mechanism design setting, each player  $i$  has a set of actions  $\mathcal{A}_i$  for  $1 \leq i \leq n$ . Given an action  $a_i \in \mathcal{A}_i$  chosen by each player  $i$  for  $1 \leq i \leq n$ , which lead to the action profile  $\mathbf{a} = (a_1, \dots, a_n) \in \mathcal{A} = \mathcal{A}_1 \times \dots \times \mathcal{A}_n$ , the auctioneer decides an outcome  $o(\mathbf{a})$  among the set of feasible outcomes  $\mathcal{O}$ . Each player  $i$  has a *private valuation* (or *type*)  $v_i$  taking values in a parameter space  $\mathcal{V}_i$ . For each outcome  $o \in \mathcal{O}$ , player  $i$  has *utility*  $u_i(o, v_i)$  depending on the outcome of the game and its valuation  $v_i$ . Since the outcome  $o(\mathbf{a})$  of the game is determined by the action profile  $\mathbf{a}$ , the utility of a player  $i$  is denoted as  $u_i(\mathbf{a}; v_i)$ . We are interested in auctions that in general consist of an allocation rule and a payment rule. Given an action profile  $\mathbf{a} = (a_1, \dots, a_n)$ , the auctioneer decides an allocation and a payment  $p_i(\mathbf{a})$  for each player  $i$ . Then, the *utility* of player  $i$  with valuation  $v_i$ , following the quasi-linear utility model, is defined as  $u_i(\mathbf{a}; v_i) = v_i - p_i(\mathbf{a})$ . The *social welfare* of an auction is defined as the total utility of all participants (the players and the auctioneer):  $\text{SW}(\mathbf{a}; \mathbf{v}) = \sum_{i=1}^n u_i(\mathbf{a}; v_i) + \sum_{i=1}^n p_i(\mathbf{a})$ .

In the paper, we consider incomplete-information settings. In contrast to the full-information settings where private valuations are deterministically determined, in incomplete-information settings the valuation vectors  $\mathbf{v}$  (in which each component is the valuation of a player) is drawn from a publicly known distribution  $\mathbf{F}$  with density function  $\mathbf{f}$ . Let  $\Delta(\mathcal{A}_i)$  be the set of probability distributions over the actions in  $\mathcal{A}_i$ . A strategy of a player is a mapping  $\sigma_i : \mathcal{V}_i \rightarrow \Delta(\mathcal{A}_i)$  from a valuation  $v_i \in \mathcal{V}_i$  to a distribution over actions  $\sigma_i(v_i) \in \Delta(\mathcal{A}_i)$ .

► **Definition 13** (Bayes-Nash equilibrium). A strategy profile  $\sigma = (\sigma_1, \dots, \sigma_n)$  is a *Bayes-Nash equilibrium* (BNE) if for every player  $i$ , for every valuation  $v_i \in \mathcal{V}_i$ , and for every action  $a'_i \in \mathcal{A}_i$ :

$$\mathbb{E}_{\mathbf{v}_{-i} \sim \mathbf{F}_{-i}(v_i)} [\mathbb{E}_{\mathbf{a} \sim \sigma(\mathbf{v})} [u_i(\mathbf{a}; v_i)]] \geq \mathbb{E}_{\mathbf{v}_{-i} \sim \mathbf{F}_{-i}(v_i)} [\mathbb{E}_{\mathbf{a}_{-i} \sim \sigma_{-i}(\mathbf{v}_{-i})} [u_i(a'_i, \mathbf{a}_{-i}; v_i)]]$$

For a vector  $\mathbf{w}$ , we use  $\mathbf{w}_{-i}$  to denote the vector  $\mathbf{w}$  with the  $i$ -th component removed. Besides,  $\mathbf{F}_{-i}(v_i)$  stands for the probability distribution over all players other than  $i$  conditioned on the valuation  $v_i$  of player  $i$ .

The price of anarchy of Bayes-Nash equilibria of an auction is defined as

$$\inf_{\mathbf{F}, \sigma} \frac{\mathbb{E}_{\mathbf{v} \sim \mathbf{F}} [\mathbb{E}_{\mathbf{a} \sim \sigma(\mathbf{v})} [\text{SW}(\mathbf{a}; \mathbf{v})]]}{\mathbb{E}_{\mathbf{v} \sim \mathbf{F}} [\text{OPT}(\mathbf{v})]}$$

where the infimum is taken over Bayes-Nash equilibria  $\sigma$  and  $\text{OPT}(\mathbf{v})$  is the optimal welfare with valuation profile  $\mathbf{v}$ .

In the paper, we consider discrete settings of valuations and payments, i.e., there are only a finite (large) number of possible valuations and payments. The main purpose of restricting to discrete settings is that we can use tools from linear programming. The continuous settings can be done by considering successively finer discrete spaces.

## 4.1 Smooth Auctions

In this section, we show that the primal-dual approach also captures the smoothness framework in studying the inefficiency of Bayes-Nash equilibria in incomplete-information settings. Smooth auctions have been defined by Roughgarden [25] and Syrgkanis and Tardos [31]. The definitions are slightly different but both are inspired by the original smoothness argument [24] and all known smoothness-based proofs can be equivalently analyzed by one of these definitions. In this section, we consider the definition of smooth auctions in [25] and revisit the price of anarchy bound of smooth auctions. In the end of the section, we show that a similar proof carries through the smooth auctions defined by Syrgkanis and Tardos [31].

► **Definition 14** ([25]). For parameters  $\lambda, \mu \geq 0$ , an auction is  $(\lambda, \mu)$ -smooth if for every valuation profile  $\mathbf{v} = (v_1, \dots, v_n)$ , there exists action distribution  $D_1^*(\mathbf{v}), \dots, D_n^*(\mathbf{v})$  over  $\mathcal{A}_1, \dots, \mathcal{A}_n$  such that, for every action profile  $\mathbf{a}$ ,

$$\sum_i \mathbb{E}_{a_i^* \sim D_i^*(\mathbf{v})} [u_i(a_i^*, \mathbf{a}_{-i}; v_i)] \geq \lambda \cdot \text{SW}(\mathbf{a}^*; \mathbf{v}) - \mu \cdot \text{SW}(\mathbf{a}; \mathbf{v}) \quad (1)$$

► **Theorem 15** ([25]). *If an auction is  $(\lambda, \mu)$ -smooth and the distributions of player valuations are independent then every Bayes-Nash equilibrium has expected welfare at least  $\frac{\lambda}{1+\mu}$  times the optimal expected welfare.*

**Proof.** Given an auction, we formulate the corresponding optimization problem by a configuration LP. A *configuration*  $A$  consists of pairs  $(i, a_i)$  such that  $(i, a_i) \in A$  means that in configuration  $A$ , player  $i$  chooses action  $a_i$ . Intuitively, a configuration is an action profile of players. For every player  $i$ , every valuation  $v_i \in \mathcal{V}_i$  and every action  $a_i \in \mathcal{A}_i$ , let  $x_{i,a_i}(v_i)$  be the variable representing the probability that player  $i$  chooses action  $a_i$ . Besides, for every valuation profile  $\mathbf{v}$ , let  $z_A(\mathbf{v})$  be the variable indicating the probability that the chosen configuration (action profile) is  $A$ . For each configuration  $A$  and valuation profile  $\mathbf{v}$ , the auctioneer outcomes an allocation and a payment and that results in a social welfare denoted as  $c_A(\mathbf{v})$ . In the other words, if  $\mathbf{a}$  is the action profile corresponding to the configuration  $A$  then  $c_A(\mathbf{v})$  is in fact  $\text{SW}(\mathbf{a}; \mathbf{v})$ . Consider the following formulation and its dual.

$$\begin{aligned} \max \quad & \sum_{\mathbf{v}} c_A(\mathbf{v}) z_A(\mathbf{v}) \\ & \sum_{a_i \in \mathcal{A}_i} x_{i,a_i}(v_i) \leq f_i(v_i) \quad \forall i, v_i \\ & \sum_A z_A(\mathbf{v}) \leq f(\mathbf{v}) \quad \forall \mathbf{v} \\ & \sum_{A: (i,a_i) \in A} z_A(v_i, \mathbf{v}_{-i}) \leq f_{-i}(\mathbf{v}_{-i}) \cdot x_{i,a_i}(v_i) \\ & \quad \quad \quad \forall i, a_i, v_i, \mathbf{v}_{-i} \\ & x_{i,a_i}(v_i), z_A(\mathbf{v}) \geq 0 \quad \forall i, a_i, A, v_i, \mathbf{v} \end{aligned} \quad \begin{aligned} \min \quad & \sum_{i, v_i} f_i(v_i) \cdot \alpha_i(v_i) + \sum_{\mathbf{v}} f(\mathbf{v}) \cdot \beta(\mathbf{v}) \\ & \alpha_i(v_i) \geq \sum_{\mathbf{v}_{-i}} f_{-i}(\mathbf{v}_{-i}) \cdot \gamma_{i,a_i}(v_i, \mathbf{v}_{-i}) \\ & \quad \quad \quad \forall i, a_i, v_i \\ & \beta(\mathbf{v}) + \sum_{(i,a_i) \in A} \gamma_{i,a_i}(\mathbf{v}) \geq c_A(\mathbf{v}) \quad \forall A, \mathbf{v} \\ & \alpha_i(v_i), \beta(\mathbf{v}), \gamma_{i,a_i}(\mathbf{v}) \geq 0 \quad \forall i, v_i, \mathbf{v} \end{aligned}$$

In the primal, the first and second constraints guarantee that variables  $x$  and  $z$  represent indeed the probability distribution of each player and the joint distribution, respectively. The third constraint makes the connection between variables  $x$  and  $z$ . It ensures that if a player  $i$  with valuation  $v_i$  selects some action  $a_i$  then in the valuation profile  $(v_i, \mathbf{v}_{-i})$ , the probability that the configuration  $A$  contains  $(i, a_i)$  must be  $f_{-i}(\mathbf{v}_{-i}) \cdot x_{i, a_i}(v_i)$ . The primal objective is the expected welfare of the auction.

**Construction of dual variables.** Assuming that the auction is  $(\lambda, \mu)$ -smooth. Fix the parameters  $\lambda$  and  $\mu$ . Given an arbitrary Bayes-Nash equilibrium  $\sigma$ , define dual variables as follows.

$$\begin{aligned}\alpha_i(v_i) &:= \frac{1}{\lambda} \mathbb{E}_{\mathbf{v}_{-i}} [\mathbb{E}_{\mathbf{b} \sim \sigma(v_i, \mathbf{v}_{-i})} [u_i(\mathbf{b}; v_i)]], \\ \beta(\mathbf{v}) &:= \frac{\mu}{\lambda} \mathbb{E}_{\mathbf{b} \sim \sigma(\mathbf{v})} [\text{SW}(\mathbf{b}; \mathbf{v})], \\ \gamma_{i, a_i}(\mathbf{v}) &:= \frac{1}{\lambda} \mathbb{E}_{\mathbf{b}_{-i} \sim \sigma_{-i}(\mathbf{v}_{-i})} [u_i(a_i, \mathbf{b}_{-i}; v_i)].\end{aligned}$$

Informally, up to some constant factors depending on  $\lambda$  and  $\mu$ ,  $\alpha_i(v_i)$  is the expected utility of player  $i$  in equilibrium  $\sigma$ ;  $\beta(\mathbf{v})$  stands for the social welfare of the auction where the valuation profile is  $\mathbf{v}$  and players follow the equilibrium actions  $\sigma(\mathbf{v})$ ; and  $\gamma_{i, a_i}(\mathbf{v})$  represents the utility of player  $i$  in valuation profile  $\mathbf{v}$  if player  $i$  chooses action  $a_i$  while other players  $i' \neq i$  follows their equilibrium strategies  $\sigma_{-i}(\mathbf{v}_{-i})$ .

**Feasibility.** We show that the constructed dual variables form a feasible solution. By the definition of dual variables, the first dual constraint reads

$$\begin{aligned}\frac{1}{\lambda} \mathbb{E}_{\mathbf{v}_{-i}} [\mathbb{E}_{\mathbf{b} \sim \sigma(\mathbf{v})} [u_i(\mathbf{b}; v_i)]] &\geq \frac{1}{\lambda} \sum_{\mathbf{v}_{-i}} f_{-i}(\mathbf{v}_{-i}) \cdot \mathbb{E}_{\mathbf{b}_{-i} \sim \sigma_{-i}(\mathbf{v}_{-i})} [u_i(a_i, \mathbf{b}_{-i}; v_i)] \\ &= \frac{1}{\lambda} \mathbb{E}_{\mathbf{v}_{-i}} [\mathbb{E}_{\mathbf{b}_{-i} \sim \sigma_{-i}(\mathbf{v}_{-i})} [u_i(a_i, \mathbf{b}_{-i}; v_i)]]\end{aligned}$$

This is exactly the definition that  $\sigma$  is a Bayes-Nash equilibrium.

For every valuation profile  $\mathbf{v} = (v_1, \dots, v_n)$  and for any configuration  $A$  (corresponding action profile  $\mathbf{a} = (a_1, \dots, a_n)$ ), the second constraint reads:

$$\frac{\mu}{\lambda} \mathbb{E}_{\mathbf{b} \sim \sigma(\mathbf{v})} [\text{SW}(\mathbf{b}; \mathbf{v})] + \sum_{(i, a_i) \in A} \frac{1}{\lambda} \mathbb{E}_{\mathbf{b}_{-i} \sim \sigma_{-i}(\mathbf{v}_{-i})} [u_i(a_i, \mathbf{b}_{-i}; v_i)] \geq \text{SW}(\mathbf{a}; \mathbf{v}). \quad (2)$$

Note that we can write  $\text{SW}(\mathbf{a}; \mathbf{v}) = \mathbb{E}_{\mathbf{b} \sim \sigma(\mathbf{v})} [\text{SW}(\mathbf{a}; \mathbf{v})]$ . For any fixed realization  $\mathbf{b}$  of  $\sigma(\mathbf{v})$ , by  $(\lambda, \mu)$ -smoothness  $\frac{\mu}{\lambda} \text{SW}(\mathbf{b}; \mathbf{v}) + \sum_i \frac{1}{\lambda} u_i(a_i, \mathbf{b}_{-i}; v_i) \geq \text{SW}(\mathbf{a}; \mathbf{v})$ . Hence, by taking expectation over  $\sigma(\mathbf{v})$ , Inequality (2) follows.

**Price of Anarchy.** The welfare of equilibrium  $\sigma$  is  $\mathbb{E}_{\mathbf{v}} \mathbb{E}_{\mathbf{b} \sim \sigma(\mathbf{v})} [\text{SW}(\mathbf{b}; \mathbf{v})]$  while the dual objective of the constructed dual variables is

$$\sum_{i, v_i} f_i(v_i) \cdot \frac{1}{\lambda} \mathbb{E}_{\mathbf{v}_{-i}} [\mathbb{E}_{\mathbf{b} \sim \sigma(\mathbf{v})} [u_i(\mathbf{b}; v_i)]] + \sum_{\mathbf{v}} f(\mathbf{v}) \cdot \frac{\mu}{\lambda} \mathbb{E}_{\mathbf{b} \sim \sigma(\mathbf{v})} [\text{SW}(\mathbf{b}; \mathbf{v})]$$

which is bounded by  $\frac{1+\mu}{\lambda} \cdot \mathbb{E}_{\mathbf{v}} \mathbb{E}_{\mathbf{b} \sim \sigma(\mathbf{v})} [\text{SW}(\mathbf{b}; \mathbf{v})]$ . Therefore, the PoA is at most  $\lambda/(1+\mu)$ . ◀



## 4.2 Simultaneous Item-Bidding Auctions

**Model.** In this section, we consider the following Bayesian combinatorial auctions. In the setting, there are  $m$  items to be sold to  $n$  players. Each player  $i$  has a private monotone valuation  $v_i : 2^{[m]} \rightarrow \mathbb{R}^+$  over different subsets of items  $S \subset 2^{[m]}$ . For simplicity, we denote  $v_i(S)$  as  $v_{iS}$ . The valuation profile  $\mathbf{v} = (v_1, \dots, v_n)$  is drawn from a *product* distribution  $\mathbf{F}$ . In other words, the probability distributions  $F_i$  of valuations  $v_i$  are independent. Designing efficient combinatorial auctions are in general complex and a major direction in literature is to seek simple and efficient auctions in term of PoA. Among others, simultaneous item-bidding auctions are of particular interest. We consider two forms of simultaneous item-bidding auctions: *simultaneous first-price auctions (S1A)* and *simultaneous second-price auctions (S2A)*. In the auctions, each player submits simultaneously a vector of bids, one for each item. A typical assumption is *non-overbidding* property in which each player submits a vector  $b_i$  of bids such that for any set of items  $S$ ,  $\sum_{j \in S} b_{ij} \leq v_{iS}$ . Given the bid profile, each item is allocated to the player with highest bid. In a simultaneous first-price auction, the payment of the winner of each item is its bid on the item; while in a simultaneous second-price auction, the winner of each item pays the second highest bid on the item.

### 4.2.1 Connection between Primal-Dual and Non-Smooth Techniques

In this section, we consider the setting in which all player valuations are *sub-additive*. That is,  $v_i(S \cup T) \leq v_i(S) + v_i(T)$  for every player  $i$  and every subsets  $S, T \subset 2^{[m]}$ . The PoA of simultaneous item-bidding auctions has been widely studied in this setting. Using smoothness framework in auctions, logarithmic bounds on PoA for S1A and S2A are given by Hassidim et al. [14] and Bhawalkar and Roughgarden [4], respectively. Recently, Feldman et al. [11] presented a significant improvement by establishing the PoA bounds 2 and 4 for S1A and S2A, respectively. Their proof arguments go beyond the smoothness framework. In the following, we revisit the results of Feldman et al. [11] and show that the duality approach captures the non-smooth technique in [11].

**Formulation.** Given a valuation profile  $\mathbf{v}$ , let  $\bar{x}_{ij}(\mathbf{v})$  be the variable indicating whether player  $i$  receives item  $j$  in valuation profile  $\mathbf{v}$ . Let  $\bar{z}_{iS}(\mathbf{v})$  be the variable indicating whether player  $i$  receives a set of items  $S$ . Then for any profile  $\mathbf{v}$  and for any item  $j$ ,  $\sum_i \bar{x}_{ij}(\mathbf{v}) \leq 1$ , meaning that an item  $j$  is allocated to at most one player. Moreover,  $\sum_{S: j \in S} \bar{z}_{iS}(\mathbf{v}) = \bar{x}_{ij}(\mathbf{v})$ , meaning that if player  $i$  receives item  $j$  then some subset of items  $S$  allocated to  $i$  must contain  $j$ . Besides,  $\sum_S \bar{z}_{iS}(\mathbf{v}) = 1$  since some subset of items (possibly empty) is allocated to  $i$ .

Let  $x_{ij}(v_i)$  and  $z_{iS}(v_i)$  be *interim* variables corresponding to  $\bar{x}_{ij}(\mathbf{v})$  and  $\bar{z}_{iS}(\mathbf{v})$  and are defined as follows:  $x_{ij}(v_i) := \mathbb{E}_{\mathbf{v}_{-i} \sim \mathbf{F}_{-i}} [\bar{x}_{ij}(v_i, \mathbf{v}_{-i})]$  and  $z_{iS}(v_i) := \mathbb{E}_{\mathbf{v}_{-i} \sim \mathbf{F}_{-i}} [\bar{z}_{iS}(v_i, \mathbf{v}_{-i})]$  where  $\mathbf{F}_{-i}$  is the product distribution of all players other than  $i$ . Consider the following relaxation with interim variables and its dual. The constraints in the primal follow the relationship between the interim variables  $x_{ij}(v_i)$ ,  $z_{iS}(v_i)$  and variables  $\bar{x}_{ij}(\mathbf{v})$ ,  $\bar{z}_{iS}(\mathbf{v})$ .

$$\begin{array}{ll}
 \max & \sum_{i,S} \sum_{v_i} f_i(v_i) [v_{iS} \cdot z_{iS}(v_i)] \\
 & \sum_i \sum_{v_i \in V_i} f_i(v_i) x_{ij}(v_i) \leq 1 \quad \forall j \\
 & \sum_S z_{iS}(v_i) = 1 \quad \forall i, v_i \\
 & \sum_{S:j \in S} z_{iS}(v_i) = x_{ij}(v_i) \quad \forall i, j, v_i \\
 & x_{ij}(v_i), z_{iS}(v_i) \geq 0 \quad \forall i, j, S, v_i \\
 \min & \sum_{i,v_i} \alpha_i(v_i) + \sum_j \beta_j \\
 & f_i(v_i) \cdot \beta_j \geq \gamma_{i,j}(v_i) \quad \forall i, j, v_i \\
 & \alpha_i(v_i) + \sum_{j \in S} \gamma_{i,j}(v_i) \geq f_i(v_i) \cdot v_{iS} \quad \forall i, S, v_i \\
 & \alpha_i(v_i) \geq 0 \quad \forall i, v_i
 \end{array}$$

**Dual Variables.** Fix a Bayes-Nash equilibrium  $\sigma$ . Given a valuation  $\mathbf{v}$ , denote  $\mathbf{b} = (b_1, \dots, b_n) = \sigma(\mathbf{v})$  as the bid equilibrium. Let  $\mathbf{B}$  be the distribution of  $\mathbf{b}$  over the randomness of  $\mathbf{v}$  and  $\sigma$ . Let  $\mathbf{B}(v_i)$  be the distribution of  $\mathbf{b}$  over the randomness of  $\mathbf{v}$  and  $\sigma$  while the valuation  $v_i$  of player  $i$  is fixed. Since  $v_i$  and  $\mathbf{v}_{-i}$  are independent and each  $\sigma_i$  is a mapping  $\mathcal{V}_i \rightarrow \Delta(\mathcal{A}_i)$ , strategy  $b_i$  is independent of  $\mathbf{b}_{-i}$ . Let  $\mathbf{B}_{-i}$  be the distribution of  $\mathbf{b}_{-i}$ . We define dual variables as follows.

Let  $\alpha_i(v_i)$  be proportional to the expected utility of player  $i$  with valuation  $v_i$ , over the randomness of valuations  $\mathbf{v}_{-i}$  of other players. Specifically,

$$\alpha_i(v_i) := 2f_i(v_i) \cdot \mathbb{E}_{\mathbf{v}_{-i} \sim \mathbf{F}_{-i}} [\mathbb{E}_{\sigma} [u_i(\sigma(v_i, \mathbf{v}_{-i}), v_i)]] = 2f_i(v_i) \cdot \mathbb{E}_{\mathbf{b} \sim \mathbf{B}(v_i)} [u_i(\mathbf{b}, v_i)]$$

Besides, let  $\gamma_{i,j}(v_i)$  be proportional to the expected value of the bid on item  $j$  if player  $i$  with valuation  $v_i$  wants to win item  $j$  while other players follow the equilibrium strategies. Formally,  $\gamma_{i,j}(v_i) := 2f_i(v_i) \cdot \mathbb{E}_{\mathbf{b}_{-i} \sim \mathbf{B}_{-i}} [\max_{k \neq i} b_{kj}]$ . Finally, define  $\beta_j := 2 \max_i \mathbb{E}_{\mathbf{b}_{-i} \sim \mathbf{B}_{-i}} [\max_{k \neq i} b_{kj}]$ .

The following lemma shows the feasibility of the variables. The main core of the proof relies on an argument in [11].

► **Lemma 16.** *The dual vector  $(\alpha, \beta, \gamma)$  defined above constitutes a dual feasible solution.*

► **Theorem 17 ([11]).** *If player valuations are sub-additive then every Bayes-Nash equilibrium of a S1A (or S2A) has expected welfare at least 1/2 (or 1/4, resp) of the optimal one.*

**Proof.** For an item  $j$ , let  $i^*(j) \in \arg \max_i \mathbb{E}_{\mathbf{v}_{-i} \sim \mathbf{F}_{-i}} [\max_{k \neq i} b_{kj}]$ . Hence,

$$\begin{aligned}
 \beta_j &= 2 \mathbb{E}_{\mathbf{v}_{-i^*(j)} \sim \mathbf{F}_{-i^*(j)}} \mathbb{E}_{\sigma} \left[ \max_{k \neq i^*(j)} b_{kj} \right] = 2 \mathbb{E}_{v_{i^*(j)} \sim F_{i^*(j)}} \mathbb{E}_{\mathbf{v}_{-i^*(j)} \sim \mathbf{F}_{-i^*(j)}} \mathbb{E}_{\sigma} \left[ \max_{k \neq i^*(j)} b_{kj} \right] \\
 &= 2 \mathbb{E}_{\mathbf{v} \sim \mathbf{F}} \mathbb{E}_{\sigma} \left[ \max_{k \neq i^*(j)} b_{kj} \right]
 \end{aligned}$$

where the second equality is due to the fact that the term  $\mathbb{E}_{\mathbf{v}_{-i^*(j)} \sim \mathbf{F}_{-i^*(j)}} \mathbb{E}_{\sigma} [\max_{k \neq i^*(j)} b_{kj}]$  is independent of  $v_{i^*(j)}$ . Therefore, the dual objective is

$$\sum_{i,v_i} \alpha_i(v_i) + \sum_j \beta_j = 2 \mathbb{E}_{\mathbf{v} \sim \mathbf{F}} \mathbb{E}_{\sigma} \left[ \sum_i u_i(\mathbf{b}, v_i) + \sum_j \max_{k \neq i^*(j)} b_{kj} \right]$$

Fix a random choice of profile  $\mathbf{v}$  and  $\sigma$  (so the bid profile  $\mathbf{b}$  is fixed). We bound the dual objective, i.e., the right-hand side of the above equality, in S1A and S2A. Note that the utility of a player winning no item is 0.

**First Price Auction.** Partition the set of items into the winning items of each player. Consider a player  $i$  with the set of winning items  $S$ . The utility of this player  $i$  is  $v_{iS} - \sum_{j \in S} \max_k b_{kj}$ . Hence,  $v_{iS} - \sum_{j \in S} b_{ij} + \sum_{j \in S} \max_{k \neq i^*(j)} b_{kj} \leq v_{iS}$  since by the allocation rule,  $b_{ij} = \max_k b_{kj}$  for every  $j \in S$ . Hence, summing over all players, the dual objective is bounded by twice the total expected valuation of winning players, which is the primal. So the price of anarchy is at most 2.

**Second Price Auction.** Similarly, consider a player  $i$  with the set of winning items  $S$ . The utility of player  $i$  as well as its payment (by no-overbidding) are at most  $v_{iS}$ . Therefore, summing over all players, the dual objective is bounded by four times the total expected valuation of winning players. Hence, the price of anarchy is at most 4. ◀

## 4.2.2 Connection between Primal-Dual and No-Envy Learning

Very recently, Daskalakis and Syrgkanis [10] have introduced *no-envy learning* – a novel concept of learning in auctions. The notion is inspired by the concept of Walrasian equilibrium and it is motivated by the fact that no-regret learning algorithms (which converge to coarse correlated equilibria) for the simultaneous item-bidding auctions are computationally inefficient as the number of player actions are exponential. When the players have fractionally sub-additive (XOS) valuation, Daskalakis and Syrgkanis [10] showed that no-envy outcomes are a relaxation of no-regret outcomes. Moreover, no-envy outcomes maintain the approximate welfare optimality of no-regret outcomes while ensuring the computational tractability. In this section, we explore the connection between the no-envy learning and the primal-dual approach. Indeed, the notion of no-envy learning would be naturally derived from the dual constraints very much in the same way as the smoothness argument is.

We recall the notion of no-envy learning algorithms [10]. We first define the *online learning problem*. In the online learning problem, at each step  $t$ , the player chooses a bid vector  $b^t = (b_1^t, \dots, b_m^t)$  where  $b_j^t$  is the bid on item  $j$  for  $1 \leq j \leq m$ ; and the adversary picks adaptively (depending on the history of the play but not on the current bid  $b^t$ ) a threshold vector  $\theta^t = (\theta_1^t, \dots, \theta_m^t)$ . The player wins the set  $S^*(b^t, \theta^t) = \{j : b_j^t \geq \theta_j^t\}$  and gets reward:

$$u(b^t, \theta^t) := v(S^*(b^t, \theta^t)) - \sum_{j \in S^*(b^t, \theta^t)} \theta_j^t$$

where  $v : 2^{[m]} \rightarrow \mathbb{R}$  is the valuation of the player.

► **Definition 18** ([10]). An algorithm for the online learning problem is *r-approximate no-envy* if, for any adaptively chosen sequence of (random) threshold vector  $\theta^{1:T}$  by the adversary, the (random) bid vector  $b^{1:T}$  chosen by the algorithm satisfies:

$$\frac{1}{T} \sum_{t=1}^T \mathbb{E}[u(b^t, \theta^t)] \geq \max_{S \subseteq [m]} \left( \frac{1}{r} \cdot v(S) - \sum_{j \in S} \frac{1}{T} \sum_{t=1}^T \mathbb{E}[\theta_j^t] \right) - \epsilon(T) \quad (3)$$

where the *no-envy rate*  $\epsilon(T) \rightarrow 0$  while  $T \rightarrow \infty$ . An algorithm is *no-envy* if it is 1-approximate no-envy.

Now we show the connection between primal-dual and no-envy learning by revisiting the following theorem. As we will see, the notion of no-envy learning corresponds exactly to a constraint of the dual program.

► **Theorem 19** ([10]). *If  $n$  players in a S2A use a  $r$ -approximate no-envy learning algorithm with envy rate  $\epsilon(T)$  then in  $T$  steps, the average welfare is at least  $\frac{1}{2r} \text{OPT} - n \cdot \epsilon(T)$  where  $\text{OPT}$  is the expected optimal welfare.*

**Proof.** Let  $b_i^t$  be the bid vector of player  $i$  where  $b_{ij}^t$  is the bid of player  $i$  on item  $j$  in step  $t$ . In a S2A the threshold  $\theta_{ij}^t = \max_{k \neq i} b_{kj}^t$ . Consider the same primal and dual LPs in Section 4.2.1.

**Dual variables.** Recall that  $r$  is the approximation factor and  $\epsilon(T)$  the no-envy rate of the learning algorithm. Define dual variables (similar to the ones in Section 4.2.1) as follows.

$$\begin{aligned} \alpha_i(v_i) &:= r \cdot f_i(v_i) \cdot \mathbb{E}_{\mathbf{v}_{-i} \sim \mathcal{F}_{-i}} \left[ \frac{1}{T} \sum_{t=1}^T \mathbb{E}_{\mathbf{b}^t(v_i, \mathbf{v}_{-i})} [u_i(b_i^t, \theta_i^t)] \right] + r \cdot \epsilon(T) \\ \gamma_{i,j}(v_j) &:= r \cdot f_i(v_i) \cdot \mathbb{E}_{\mathbf{v}_{-i} \sim \mathcal{F}_{-i}} \left[ \frac{1}{T} \sum_{t=1}^T \mathbb{E}_{\mathbf{b}^t(v_i, \mathbf{v}_{-i})} [\theta_{ij}^t] \right] = r \cdot f_i(v_i) \cdot \mathbb{E}_{\mathbf{v}_{-i} \sim \mathcal{F}_{-i}} \left[ \frac{1}{T} \sum_{t=1}^T \mathbb{E}_{\mathbf{b}_{-i}^t(\mathbf{v}_{-i})} [\theta_{ij}^t] \right] \\ \beta_j &:= r \cdot \max_i \max_{v_i} \mathbb{E}_{\mathbf{v}_{-i} \sim \mathcal{F}_{-i}} \left[ \frac{1}{T} \sum_{t=1}^T \mathbb{E}_{\mathbf{b}^t(v_i, \mathbf{v}_{-i})} [\theta_{ij}^t] \right] = r \cdot \max_i \mathbb{E}_{\mathbf{v}_{-i} \sim \mathcal{F}_{-i}} \left[ \frac{1}{T} \sum_{t=1}^T \mathbb{E}_{\mathbf{b}_{-i}^t(\mathbf{v}_{-i})} [\theta_{ij}^t] \right] \end{aligned}$$

where the second equalities in the definitions of  $\gamma$  and  $\beta$  follow the fact that player valuations are independent and  $\theta_{ij}^t$  does not depend on  $b_{ij}^t$  for every  $i, j$ .

**Feasibility.** The first dual constraint follows immediately by the definitions of dual variables  $\beta$  and  $\gamma$ . For a fixed set  $S$  and a player  $i$  with valuation  $v_i$ , the second dual constraint reads

$$\begin{aligned} r \cdot f_i(v_i) \cdot \mathbb{E}_{\mathbf{v}_{-i} \sim \mathcal{F}_{-i}} \left[ \frac{1}{T} \sum_{t=1}^T \mathbb{E}_{\mathbf{b}^t(v_i, \mathbf{v}_{-i})} [u_i(b_i^t, \theta_i^t)] \right] + r \cdot \epsilon(T) \\ + r \cdot \sum_{j \in S} f_i(v_i) \cdot \mathbb{E}_{\mathbf{v}_{-i} \sim \mathcal{F}_{-i}} \left[ \frac{1}{T} \sum_{t=1}^T \mathbb{E}_{\mathbf{b}_{-i}^t(\mathbf{v}_{-i})} [\theta_{ij}^t] \right] \geq f_i(v_i) \cdot v_{iS} \end{aligned}$$

This inequality follows immediately from the definition of  $r$ -approximate no-envy learning algorithms (specifically, Inequality (3)) by simplifying and rearranging terms. (Note that  $\mathbb{E}_{\mathbf{v}_{-i} \sim \mathcal{F}_{-i}} [f_i(v_i) \cdot v_{iS}] = f_i(v_i) \cdot v_{iS}$ ).

**Bounding the cost.** In  $T$  steps, the average welfare is

$$\mathbb{E}_{\mathbf{v}} \left[ \frac{1}{T} \sum_{t=1}^T \mathbb{E}_{\mathbf{b}^t(\mathbf{v})} [v_i(b_i^t, \theta_i^t)] \right] = \mathbb{E}_{\mathbf{v}} \left[ \frac{1}{T} \sum_{t=1}^T \mathbb{E}_{\mathbf{b}^t(\mathbf{v})} [v_i(S^*(b_i^t, \theta_i^t))] \right].$$

Besides, in the dual objective,

$$\begin{aligned} \sum_{i, v_i} \alpha_i(v_i) &\leq r \cdot \mathbb{E}_{\mathbf{v}} \left[ \frac{1}{T} \sum_{t=1}^T \mathbb{E}_{\mathbf{b}^t(\mathbf{v})} [v_i(S^*(b_i^t, \theta_i^t))] \right] + n \cdot r \cdot \epsilon(T), \\ \sum_j \beta_j &\leq r \cdot \mathbb{E}_{\mathbf{v}} \left[ \frac{1}{T} \sum_{t=1}^T \mathbb{E}_{\mathbf{b}^t(\mathbf{v})} [v_i(S^*(b_i^t, \theta_i^t))] \right] \end{aligned}$$

where the last inequality is due to the non-overbidding property. Hence, the theorem follows by weak duality. ◀

## 5 Conclusion

In the paper, we have presented a primal-dual approach to study the efficiency of games. We have shown the applicability of the approach on a wide variety of settings and have given simple and improved analyses for several problems in settings of different natures. Beyond concrete results, the main point of the paper is to illuminate the potential of the primal-dual approach. In this approach, the PoA-bound analyses now can be done similarly as the analyses of LP-based algorithms in Approximation/Online Algorithms. We hope that linear programming and duality would bring new ideas and techniques, from well-developed domains such as approximation, online algorithms, etc to algorithmic game theory, not only for the analyses and the understanding of current games but also for the design of new games (auctions) and new concepts leading to improved efficiency.

---

## References

- 1 Robert J Aumann. Subjectivity and correlation in randomized strategies. *Journal of mathematical Economics*, 1(1):67–96, 1974.
- 2 Kshipra Bhawalkar, Martin Gairing, and Tim Roughgarden. Weighted congestion games: the price of anarchy, universal worst-case examples, and tightness. *ACM Transactions on Economics and Computation*, 2(4):14, 2014.
- 3 Kshipra Bhawalkar, Sreenivas Gollapudi, and Kamesh Munagala. Coevolutionary opinion formation games. In *Proceedings of the forty-fifth annual ACM symposium on Theory of computing*, pages 41–50. ACM, 2013.
- 4 Kshipra Bhawalkar and Tim Roughgarden. Welfare guarantees for combinatorial auctions with item bidding. In *Proceedings of the twenty-second annual ACM-SIAM symposium on Discrete Algorithms*, pages 700–709. SIAM, 2011.
- 5 Vittorio Bilo. A unifying tool for bounding the quality of non-cooperative solutions in weighted congestion games. In *International Workshop on Approximation and Online Algorithms*, pages 215–228, 2012.
- 6 Niv Buchbinder and Joseph Naor. The design of competitive online algorithms via a primal-dual approach. *Foundations and Trends® in Theoretical Computer Science*, 3(2–3):93–263, 2009.
- 7 Ioannis Caragiannis, Christos Kaklamanis, Panagiotis Kanellopoulos, Maria Kyropoulou, Brendan Lucier, Renato Paes Leme, and Eva Tardos. Bounding the inefficiency of outcomes in generalized second price auctions. *Journal of Economic Theory*, 156:343–388, 2015.
- 8 Roberto Cominetti, José R Correa, and Nicolás E Stier-Moses. The impact of oligopolistic competition in networks. *Operations Research*, 57(6):1421–1437, 2009.
- 9 José R Correa, Andreas S Schulz, and Nicolás E Stier-Moses. A geometric approach to the price of anarchy in nonatomic congestion games. *Games and Economic Behavior*, 64(2):457–469, 2008.
- 10 Constantinos Daskalakis and Vasilis Syrgkanis. Learning in auctions: Regret is hard, envy is easy. In *57th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 219–228, 2016.
- 11 Michal Feldman, Hu Fu, Nick Gravin, and Brendan Lucier. Simultaneous auctions are (almost) efficient. In *Proceedings of the forty-fifth annual ACM symposium on Theory of computing*, pages 201–210. ACM, 2013.
- 12 Michal Feldman, Nicole Immorlica, Brendan Lucier, Tim Roughgarden, and Vasilis Syrgkanis. The price of anarchy in large games. In *Proc. 48th Symposium on Theory of Computing (STOC)*, pages 963–976, 2016.

- 13 Tobias Harks. Stackelberg strategies and collusion in network games with splittable flow. *Theory of Computing Systems*, 48(4):781–802, 2011.
- 14 Avinatan Hassidim, Haim Kaplan, Yishay Mansour, and Noam Nisan. Non-price Equilibria in Markets of Discrete Goods. In *Proc. 12th ACM Conference on Electronic Commerce*, pages 295–296, 2011.
- 15 Elias Koutsoupias and Christos Papadimitriou. Worst-case equilibria. *Computer science review*, 3(2):65–69, 2009.
- 16 Vijay Krishna. *Auction theory*. Academic press, 2009.
- 17 Janardhan Kulkarni and Vahab Mirrokni. Robust price of anarchy bounds via LP and fenchel duality. In *Proceedings of the Twenty-Sixth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1030–1049. SIAM, 2015.
- 18 Renato Paes Leme, Vasilis Syrgkanis, and Éva Tardos. Sequential auctions and externalities. In *Proceedings of the twenty-third annual ACM-SIAM symposium on Discrete Algorithms*, pages 869–886. SIAM, 2012.
- 19 Giorgio Lucarelli, Nguyen Kim Thang, Abhinav Srivastav, and Denis Trystram. Online Non-preemptive Scheduling in a Resource Augmentation Model based on Duality. In *European Symposium on Algorithms*, 2016.
- 20 Konstantin Makarychev and Maxim Sviridenko. Solving optimization problems with diseconomies of scale via decoupling. In *Foundations of Computer Science (FOCS), 2014 IEEE 55th Annual Symposium on*, pages 571–580. IEEE, 2014.
- 21 Hervé Moulin and J-P Vial. Strategically zero-sum games: the class of games whose completely mixed equilibria cannot be improved upon. *International Journal of Game Theory*, 7(3-4):201–221, 1978.
- 22 Uri Nadav and Tim Roughgarden. The limits of smoothness: A primal-dual framework for price of anarchy bounds. In *International Workshop on Internet and Network Economics*, pages 319–326, 2010.
- 23 John F Nash. Equilibrium points in n-person games. *Proc. Nat. Acad. Sci. USA*, 36(1):48–49, 1950.
- 24 Tim Roughgarden. Intrinsic robustness of the price of anarchy. *Journal of the ACM (JACM)*, 62(5):32, 2015.
- 25 Tim Roughgarden. The price of anarchy in games of incomplete information. *ACM Transactions on Economics and Computation*, 3(1):6, 2015.
- 26 Tim Roughgarden and Florian Schoppmann. Local smoothness and the price of anarchy in splittable congestion games. *Journal of Economic Theory*, 156:317–342, 2015.
- 27 Tim Roughgarden, Vasilis Syrgkanis, and Eva Tardos. The Price of Anarchy in Auctions. *Journal of Artificial Intelligence Research*, 59:59–101, 2017.
- 28 Tim Roughgarden and Éva Tardos. How bad is selfish routing? *Journal of the ACM (JACM)*, 49(2):236–259, 2002.
- 29 Tim Roughgarden and Éva Tardos. Bounding the inefficiency of equilibria in nonatomic congestion games. *Games and Economic Behavior*, 47(2):389–403, 2004.
- 30 Vasilis Syrgkanis and Eva Tardos. Bayesian sequential auctions. In *Proceedings of the 13th ACM Conference on Electronic Commerce*, pages 929–944. ACM, 2012.
- 31 Vasilis Syrgkanis and Eva Tardos. Composable and efficient mechanisms. In *Proceedings of the forty-fifth annual ACM symposium on Theory of computing*, pages 211–220. ACM, 2013.
- 32 Nguyen Kim Thang. Game Efficiency through Linear Programming Duality. *CoRR*, 2017. [arXiv:1708.06499](https://arxiv.org/abs/1708.06499).
- 33 Rakesh V Vohra. *Mechanism design: a linear programming approach*, volume 47. Cambridge University Press, 2011.
- 34 David P Williamson and David B Shmoys. *The design of approximation algorithms*. Cambridge university press, 2011.